# NAVAL POSTGRADUATE SCHOOL
## Monterey, California

# THESIS

**THE DESIGN AND DEVELOPMENT OF
A WEB-INTERFACE FOR THE SOFTWARE
ENGINEERING AUTOMATION SYSTEM**

by

James A. McDonald III

September 2001

| | |
|---|---|
| Thesis Advisor: | Man-Tak Shing |
| Co-Advisor: | Richard Riehle |

# Report Documentation Page

| Report Date | Report Type | Dates Covered (from... to) |
|---|---|---|
| 30 Sep 2001 | N/A | - |

| Title and Subtitle | Contract Number |
|---|---|
| The Design and Development of a Web Interface for the Software Engineering Automation System | Grant Number |
| | Program Element Number |

| Author(s) | Project Number |
|---|---|
| Major James A. McDonald III | Task Number |
| | Work Unit Number |

| Performing Organization Name(s) and Address(es) | Performing Organization Report Number |
|---|---|
| Research Office Naval Postgraduate School Monterey, Ca 93943-5138 | |

| Sponsoring/Monitoring Agency Name(s) and Address(es) | Sponsor/Monitor's Acronym(s) |
|---|---|
| | Sponsor/Monitor's Report Number(s) |

**Distribution/Availability Statement**
Approved for public release, distribution unlimited

**Supplementary Notes**

**Abstract**

**Subject Terms**

| Report Classification | Classification of this page |
|---|---|
| unclassified | unclassified |

| Classification of Abstract | Limitation of Abstract |
|---|---|
| unclassified | UU |

**Number of Pages**
241

| REPORT DOCUMENTATION PAGE | | | *Form Approved OMB No. 0704-0188* |
|---|---|---|---|
| Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503. | | | |
| **1. AGENCY USE ONLY** *(Leave blank)* | **2. REPORT DATE** September 2001 | **3. REPORT TYPE AND DATES COVERED** Master's Thesis | |
| **4. TITLE AND SUBTITLE** The Design and Development of a Web Interface for the Software Engineering Automation System | | **5. FUNDING NUMBERS** | |
| **6. AUTHOR (S)** Major James A. McDonald III | | | |
| **7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)** Naval Postgraduate School Monterey, CA 93943-5000 | | **8. PERFORMING ORGANIZATION REPORT NUMBER** | |
| **9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)** | | **10. SPONSORING/MONITORING AGENCY REPORT NUMBER** | |
| **11. SUPPLEMENTARY NOTES** The views expressed in this thesis are those of the author and do not reflect the official policy or position of the U.S. Department of Defense or the U.S. Government. | | | |
| **12a. DISTRIBUTION / AVAILABILITY STATEMENT** Approved for public release; distribution is unlimited | | **12b. DISTRIBUTION CODE** | |

**13. ABSTRACT** *(maximum 200 words)*

The Software Engineering Automation System evolved from the Computer-Aided Prototyping System (CAPS) developed in the late 1980's and early 1990's to help software engineers rapidly produce working prototypes for hard real-time embedded systems. As software development methods such as the waterfall and spiral methods evolved the requirement for a system to prototype products became clear. CAPS was able to meet the needs of the software engineer, allowing them to edit the project, translate and compile the code, develop the interface, and execute the project. As the requirements change and customer's needs become clearer, the ability to rapidly change the prototype to meet these needs was met by the CAPS system. Today companies that are developing software systems are global in nature. Development could take place over a vast expanse of several continents. The change in the workplace environment bore the requirement to redesign SEAS to make it accessible globally as well as making it functional across multiple platforms. The envisioned redesign of the SEAS system takes the functionality of the current system and deploys it as a web application on the Internet.

| **14. SUBJECT TERMS** Computer Aided Prototyping, Real Time Systems, Java | | | **15. NUMBER OF PAGES** 241 |
|---|---|---|---|
| | | | **16. PRICE CODE** |
| **17. SECURITY CLASSIFICATION OF REPORT** Unclassified | **18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified | **19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified | **20. LIMITATION OF ABSTRACT** UL |

THIS PAGE INTENTIONALLY LEFT BLANK

THE DESIGN AND DEVELOPMENT OF
A WEB-INTERFACE FOR THE SOFTWARE
ENGINEERING AUTOMATION SYSTEM

James A. McDonald III
Major, United States Marine Corps
B.S., Virginia Military Institute, 1986

Submitted in partial fulfillment of the
requirements for the degree of
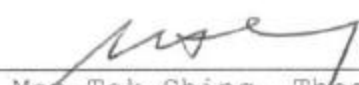
MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
September 2001

Author:  _____
         James A. McDonald III

Approved by:  _____
              Man-Tak Shing, Thesis Advisor

              _____
              Richard Riehle, Co-Advisor

              _____
              C.S. Eagle, Chairman
              Department of Computer Science

iii

THIS PAGE INTENTIONALLY LEFT BLANK

**ABSTRACT**

The Software Engineering Automation System evolved from the Computer-Aided Prototyping System (CAPS) developed in the late 1980's and early 1990's to help software engineers rapidly produce working prototypes for hard real-time embedded systems. As software development methods such as the waterfall and spiral methods evolved the requirement for a system to prototype products became clear. CAPS was able to meet the needs of the software engineer, allowing them to edit the project, translate and compile the code, develop the interface, and execute the project. As the requirements change and customer's needs become clearer, the ability to rapidly change the prototype to meet these needs was met by the CAPS system. Today companies that are developing software systems are global in nature. Development could take place over a vast expanse of several continents. The change in the workplace environment bore the requirement to redesign SEAS to make it accessible globally as well as making it functional across multiple platforms. The envisioned redesign of the SEAS system takes the functionality of the current system and deploys it as a web application on the Internet.

THIS PAGE INTENTIONALLY LEFT BLANK

# TABLE OF CONTENTS

# LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

# LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

## ACKNOWLEDGEMENTS

THIS PAGE INTENTIONALLY LEFT BLANK

# I. INTRODUCTION

Over the past three decades the development of software has given rise to several models that depict how software is developed from start to finish. All were similar to a point in that they covered the categories of requirements analysis, specifications, design issues, coding and testing. Of the categories mentioned above perhaps the most important is the first: Requirements. Requirements embody what the customer is looking for in the functionality of the product. Through interaction with the customer, the software engineer extracts from the customer's stated needs the requirements of the software. All too often the mapping of the desires of the customer to the requirements and implementation of the software by the software engineer does not match up. Some needs are not met correctly and some needs are not included in the final product. The reasons for these mismatched requirements versus customers needs are many. Perhaps there is a communication gap or more often than not it is merely a difference of interpretation from the user's environment to the developer's view of the user's environment. Unfortunately, the results of such mismatch are often discovered at the end of the coding phase of development. The software engineer delivers his product to the customer and the customer finds the product does not function as he had envisioned. Or according to Leffingwell and Widrig in *Managing Software Requirement: A Unified Approach* the "Yes but…" and "Undiscovered Ruins" Syndromes occur.[8] That is the customer finds the final product is not exactly what he

1

meant, or now that he sees the final product he has another requirement to add.  At this point in development both the developer and the customer have wasted precious money and resources.  One method available to avoid this problem is the idea of prototyping.

## A.   BACKGROUND OF PROTOTYPING

Prototyping is merely developing a scaled down working version of the final product that the developer can present to the customer for evaluation.  The prototype is developed after the creation of the requirements and specifications. The customer can evaluate the prototype and provide the feedback necessary to the developer to determine if the requirements map correctly to the customer's desires. Adding additional requirements, or changing requirements can be done cheaply and efficiently at this stage of development.  When the idea of prototyping was first incorporated into the development cycle the software engineer was required to develop the prototype manually. Making changes to the prototype based on changes requested by the customer was difficult and time consuming.  Several iterations of changes to the prototype could be very time consuming to the developer and perhaps more trouble than it was worth.  Prototyping has become more feasible with the advent of automated tools developed to generate the code necessary depending on the requirements.[1]

One field of software development that benefits greatly from prototyping is real-time embedded systems. These systems are concerned with time schedules, input and output variables and target languages.  Being able to model a proposed system based on a user's given requirements can

be very difficult. Developing the entire system only to find it does not meet the customer's needs is a tremendous waste of time. Real-time systems need a special type of prototyping to allow for improved design of software systems. The prototyping needed is rapid software prototyping.[2] Rapid software prototyping is an iterative software development methodology utilized to improve the analysis and design of real-time systems [2]. One software development tool available is the Software Engineering Automation System (SEAS). This tool, developed by the Naval Postgraduate School, Software Engineering Group, enables the developer to rapidly produce working prototypes of real-time systems.

## B. EVOLUTION OF THE CAPS SYSTEM

The original version of SEAS evolved from CAPS, the Computer Aided Prototyping System which was developed using the C and Ada programming languages for implementation in a UNIX environment. CAPS is an integrated collection of tools that are linked together by a user interface to form a software development environment. CAPS consisted of a subsystem of editors, an execution support subsystem, and a software base subsystem. The editor subsystem was made up of a Prototype Software Development Language (PSDL) editor, an Ada editor, and an interface editor. The execution support subsystem contained a translator, a scheduler, and a compiler. The software base subsystem is part of a software database system. CAPS provides reusable software components for each prototype previously developed in CAPS that has a complete PSDL specification and executable code. The software base is capable of keeping track of the PSDL

3

description and Ada implementations for all reusable software components in CAPS.

As software development evolved into global cooperative efforts with the explosion of the Internet, the need for CAPS to support a geographically dispersed team of designers and customers became critical. The need for CAPS to allow users to model, develop, execute and evaluate prototypes of proposed systems from dispersed locations utilizing different hardware platforms and operating environments became prevalent. This thesis is concerned with developing a web interface for SEAS allowing users to utilize SEAS over the Internet to model, develop and execute prototypes.

Over the past five years CAPS has slowly begun a transition from the once UNIX based system to the SEAS system capable of running on multiple platforms to include Linux and Microsoft Windows utilizing the portability of the Java programming language. The most significant step was the development of the PSDL editor using Java 1.1 and subsequently its upgrade to Java versions 1.2.2 and 1.3. This was the first step in meeting a SEAS system which could be fully ported to any one of the three major operating systems: Microsoft Windows, UNIX, and Linux. Once developed, the Java PSDL Editor was used to develop a working prototype. The PSDL code generated from the editor was then translated, scheduled and compiled using the original CAPS UNIX systems. The Ada source code generated by the compilation was then compiled using JGNAT 1.1 compiler. JGNAT translated the Ada source code to Java classes. A simple graphical user interface was developed

4

using Java's inherent classes and tied to the JGNAT compiled Java classes. The result was a prototype fully executable on all three platforms. This proof of concept paved the way for the re-design of the CAPS System to take advantage of the Java programming language and to proceed with implementation of SEAS via the Internet.

## C. THESIS OBJECTIVES

The objective of this thesis is to re-design and implement the CAPS interface as a web application. In so doing this implementation will be a step toward developing a truly distributed design tool for real-time embedded systems. The functionality of the system will be identical to the current UNIX based CAPS system. The projects developed by users of the system will be exclusively controlled by those users. The proposed system only supports single user prototyping model. Additional configuration management and distributed concurrent team development control will be needed to make it into a truly distributed prototyping environment. To the user the behavior of the system will be transparent. They will have access to the same tools as are available with the current CAPS system.

The requirements for this web-based application were based on the original requirements of CAPS. In most web-based applications it is common to find that the system itself is executing centrally on a server with a web browser acting only as an interface for the user to the system. The requirements of the system had to take this into consideration when developing the architecture as well as the implementation of the system. The final

5

consideration will be the decision to reuse as much of the current CAPS system as possible. Each aspect of the existing CAPS system was evaluated as well as previously developed CAPS tools, such as the Java-based editor, for its usefulness in the newly developed system. In the end the usability and capabilities of a web-based SEAS will determine whether it is a success and worth the effort to expand upon its development.

## II.  REQUIREMENTS ANALYSIS FOR THE REDESIGN OF CAPS AS A WEB APPLICATION

Perhaps the most important part of any system's design is the development of the requirements.  If the requirements are incorrect the resulting product will in turn be incorrect.  We had one advantage in the development of requirements for this system:  a working version of CAPS was already deployed and functioning.  Therefore our efforts concentrated on mapping the execution of the current UNIX based CAPS system into simple requirements that could then be used to re-implement the system in a web-based architecture.

### A.  METHOD OF MODELING THE SYSTEM

The method chosen to model the web interface was the Unified Modeling Method (UML).  UML is a flexible and simple tool that can be used to model software systems.  It is graphical in nature making it easy to understand and visualize.  Visualization was key for us in the re-design of this system.  UML itself can be used from the outset of a project to the end stages of the project.  It allows the designers, stakeholder, managers and others to provide input into the system design.  It can then be used to develop requirements, documentation, and source code.  Our use of UML was limited to the development of requirements, documentation and visualization of the system.  In future developments and enhancements of the system developers will be able to use the generated UML artifacts as a starting point to understanding the design of the system.

7

The first step in developing the requirements was the determination of the system's functions that were absolutely necessary in order to implement the system on the web. These system functions are those functions that the system must perform. Because this was a re-design of the original CAPS all currently implemented functional requirements were a must for web-based SEAS. Anything less would be unsatisfactory and consequently a failure. Therefore the majority of the requirements were listed as a "must" for the constraints of the system.

Once the system functions were identified they were depicted using UML use cases diagrams. A use case is a set of scenarios tied together by a common user goal.[9] The use cases developed from the use case diagram are in a narrative format and describe the interaction between the system and its surroundings.[9] Properly utilized they will capture the behavior necessary for the system to perform. Use case diagrams allow the developer to step back and view his system ensuring all the parts are included and the appropriate behavior is modeled. Each use case diagram together with their extended use case can then be utilized to generate sequence diagrams. A UML sequence diagram depicts a particular function of the system from start to finish. UML allows the developer at this point to begin to fill in the missing pieces of the model to actually bring the model a step closer to the final project. Classes and methods can now be applied at this early stage.

## B. REQUIREMENTS ANALYSIS RESULTS

The results of the requirements analysis for redesigning SEAS as a web-based system are detailed below.

### 1. Functional Requirements

The functional requirements for the redesign of SEAS are listed in Tables 2.1 – 2.4 grouped by system modeling, system administration, network support, and project management.

| Ref# | Function | Function Category | Attribute | Details and Constraints | Detail Category |
|------|----------|-------------------|-----------|-------------------------|-----------------|
| R1.1 | Model Hard Real Time Systems | Evident | Interface Metaphor | Graphical user interface for flow diagram and form metaphor input for hard real time constraints | Must |
| R1.2 | Launch the CAPS Editor | Evident | Ease of use | System must invoke the CAPS Editor and load the project set on the server | Must |
| R1.3 | Upload Files | Evident | Ease of use | System must be able to upload files from the local machine to the users server workspace | Must |
| R1.4 | Download Files | Evident | Ease of use | System must be able to download files from the users server folder to the users client system | Must |
| R1.5 | Translate Project | Evident | Ease of use | System must be able to invoke the CAPS translator on the set project | Must |
| R1.6 | Schedule Project | Evident | Ease of use | System must be able to invoke the CAPS scheduler on the set project | Must |
| R1.7 | Compile Project | Evident | Ease of use | System must be able to invoke the CAPS compiler on the set | Must |

| Ref# | Function | Function Category | Attribute | Details and Constraints | Detail Category |
|---|---|---|---|---|---|
| | | | | project and create an executable | |
| R1.8 | Execute Project | Evident | Ease of use | System must be able to invoke the project executable and be viewed locally by the user using either LINUX, UNIX, or Windows | Must |

Table 1. User Input Function – System Modeling/Specification and Prototype Development

| Ref# | Function | Function Category | Attribute | Details and Constraints | Detail Category |
|---|---|---|---|---|---|
| R2.1 | Allow system administrator to add new users to the system | Hidden | Scalability | Allow for a max of 50 users | Must |
| R2.2 | Allow system administrator to delete users from the system | Hidden | Scalability | Allow for a max of 50 users | Must |
| R2.3 | Allow system administrator to update user in the system | Hidden | Security | Ability to change user account information | Must |
| R2.4 | Allow system administrator to recover from a server failure | Hidden | Fault tolerance | Ability to restart the system as well as recover from a storage failure | Must |
| R2.5 | Allow system administrator to access the DBMS | Hidden | Security | Ability of administrator to start and restart the DBMS | Must |

Table 2. System Administration

| Ref# | Function | Function Category | Attribute | Details and Constraints | Detail Category |
|---|---|---|---|---|---|
| R3.1 | Transmit information via HTML to the user | Evident | Interface | Must be able to generate html files to the user for system status | Must |
| R3.2 | Output html files to the user | Evident | File access | Must be able to transfer static and dynamically generated files to the user | Must |
| R3.3 | Upload files to the server | Evident | File access | Must be able to transfer project files from the user client machine | Must |

| | | | | to the user server project folder for use in development | |
|---|---|---|---|---|---|
| R3.4 | Communicate with database | Evident | Database access | Must be able to access the system database in order to update or receive information concerning a users projects as well as system administration | Must |

Table 3.          Network Support

| Ref# | Function | Function Category | Attribute | Details and Constraints | Detail Category |
|---|---|---|---|---|---|
| R4.1 | User must login to use system | Evident | Security | Must be able to determine if user is authorized to use the system or not | Must |
| R4.2 | Create a new project | Evident | Response time | Less than 20 sec | Must |
| R4.3 | Create a new project version | Evident | Response time | Less that 20 sec | Must |
| R4.4 | Set the project to be developed in the system | Evident | | Must be able to keep track of which project the user is working on at any one time | Must |

Table 4.          Project Management

## 2.  Use Case Diagrams

The use case diagrams below were developed using Rational Rose Enterprise Suite 2001.  They depict both the user and administrator systems.
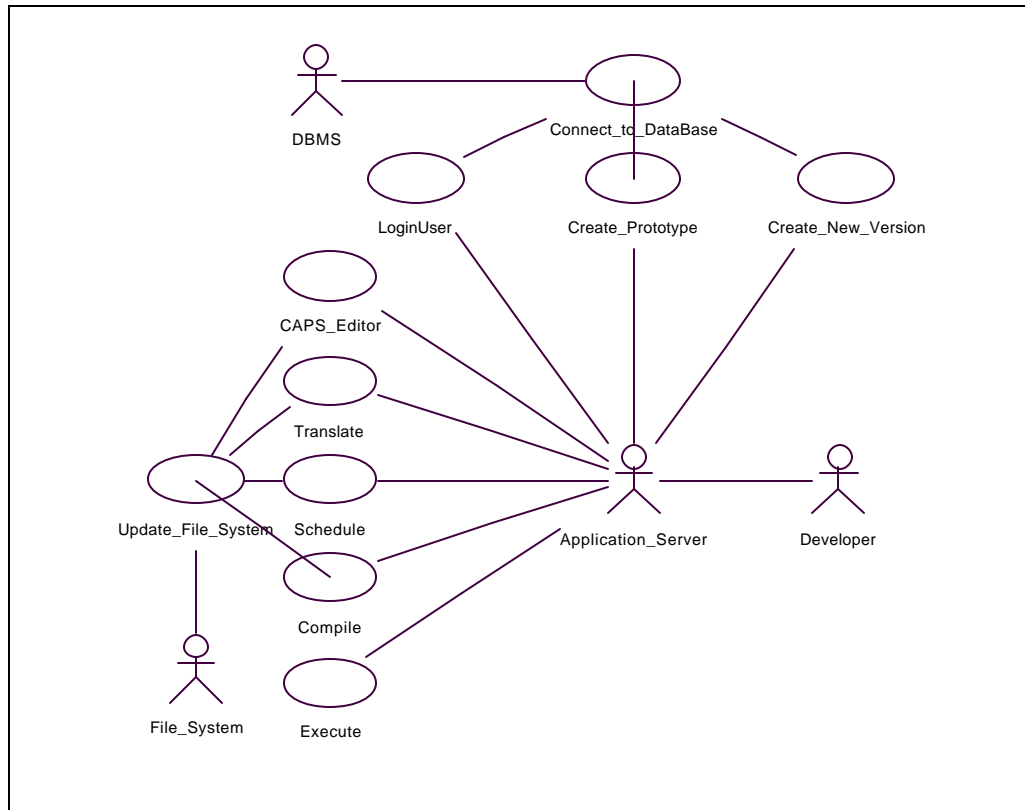
Figure 1.        Use Case Diagram User System
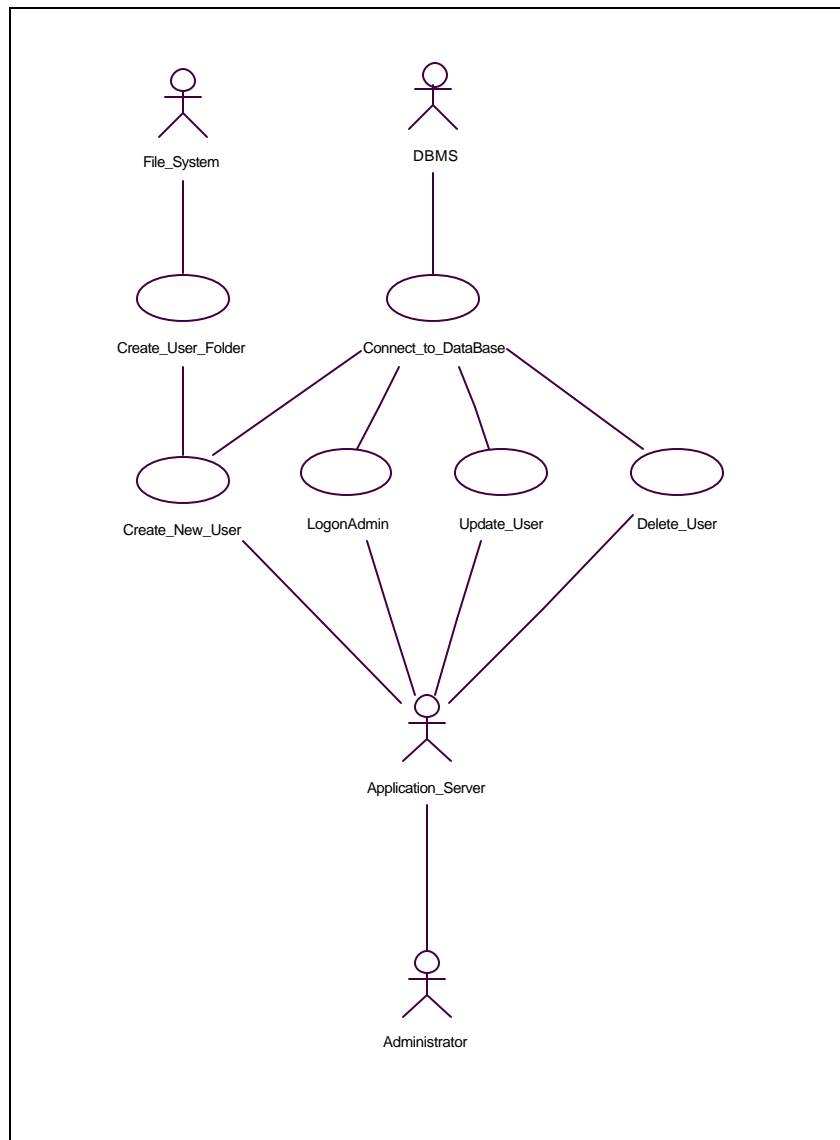
Figure 2.        Use Case Diagram System Administrator

### 3. Extended Use Cases

The extended use cases are detailed below.

| ITEM | VALUE |
|---|---|
| Use Case Name | Create a project |
| Brief Description | This use case describes the actions which will be performed by the client in order to create a new project |
| Flow of Events | The user will execute the "create new project" hyperlink. This will then allow the user to specify a project name and version.  The application server will check to ensure it is a new project and if so will update the database |

13

| ITEM | VALUE |
| --- | --- |
| | and create a project folder within the users workspace |
| Alternate Flow of Events | User denied access caused by unauthorized user trying to access secure site.<br>Project already exists |
| Pre-conditions | User must have a user account<br>Client must be logged on to the system<br>Project must not already be created |
| Post-condition | Database updated with new project/version name<br>Folder created within the user workspace |
| Special Requirements | N/A |

| ITEM | VALUE |
| --- | --- |
| Use Case Name | Log-in |
| Brief Description | This use case describes the actions that will be performed by the application server to log a client on to the system. |
| Flow of Events | A message is received from a client with a user name and a password.  Application server will pass the information to the database server for verification of the information.  If correct web server will serve the web site to the user.<br>If incorrect a dialogue message will inform the user. |
| Alternate Flow of Events | N/A |
| Pre-conditions | Application server is running and connected to internet<br>Database server is running and connected to application server to allow communication |
| Post-condition | User is logged in to system and allowed to proceed |
| Special Requirements | N/A |

| ITEM | VALUE |
| --- | --- |
| Use Case Name | Log-off user |
| Brief Description | This use case describes the actions that will be performed by the application server when a user logs off the system. |
| Flow of Events | User will activate the log-off hyperlink.<br>Application server will destroy the session and return the user to the login page. |
| Alternate Flow of Events | N/A |
| Pre-conditions | User must be logged on system |
| Post-condition | Application server serves up a page confirming to the user that he/she is indeed logged off. |
| Special Requirements | N/A |

| ITEM | VALUE |
| --- | --- |
| Use Case Name | Add user |
| Brief Description | Describes the actions needed for an administrator to add a user to the database. |
| Flow of Events | Administrator will log-in as a system administrator via the Internet.<br>Using an html form interface the administrator will create an account for the user.<br>The information will be processes by the application server and passed to the database.<br>If it is a unique username then an account will be created within the database and a workspace folder will be created for the user |

| Alternate Flow of Events | Command line entry of information directly to the database using SQL |
|---|---|
| Pre-conditions | User not already added to database. Communication b/n application server and database |
| Post-condition | Users account is activated and database is updated. |
| Special Requirements | N/A |

| ITEM | VALUE |
|---|---|
| Use Case Name | Use CAPS Editor |
| Brief Description | This use case describes the actions that will be performed by the application server to allow the user to utilize the CAPS Editor |
| Flow of Events | The user will execute the "launch editor" hyperlink on the html interface page.  This will prompt the application server to launch the CAPS Editor on the client machine with the information needed for the current project/version.  The user will then be able to edit the project using the PSDL Editor |
| Alternate Flow of Events | N/A |
| Pre-conditions | User is logged on to the system. The user has specified a project/version to work on |
| Post-condition | CAPS Editor Application is running on the system. |
| Special Requirements | N/A |

| ITEM | VALUE |
|---|---|
| Use Case Name | Translate |
| Brief Description | This use case describes the actions which will be performed by the application when the user wishes to translate the psdl file of a project |
| Flow of Events | When the "translate" hyperlink is activated then the server will invoke the CAPS translator.  The server will return to the client a successful or not-successful dialog. |
| Alternate Flow of Events | If service is unavailable at this time then a message to that affect will be delivered. |
| Pre-conditions | User logged on to the system. The project/version must be set by the user The server has all files available for the project in order to translate it. |
| Post-condition | User either has a successful or unsuccessful translation |
| Special Requirements | N/A |

| ITEM | VALUE |
|---|---|
| Use Case Name | Schedule |
| Brief Description | This use case describes the actions which will be performed by the application when the user wishes to schedule the project |
| Flow of Events | When the "schedule" hyperlink is activated then the server will invoke the CAPS scheduler.  The server will return to the client a successful or not-successful dialog.  If successful the user will be able to view the schedule |
| Alternate Flow of Events | If service is unavailable at this time then a message to that affect will be delivered. |
| Pre-conditions | User logged on to the system. The project/version must be set by the user The server has all files available for the project in order to schedule the project ie:  Must have a successful translation. |

15

| ITEM | VALUE |
|---|---|
| Post-condition | User either has a successful or unsuccessful schedule. If successful then a schedule is viewable |
| Special Requirements | N/A |

| ITEM | VALUE |
|---|---|
| Use Case Name | Compile |
| Brief Description | This use case describes the actions which will be performed by the application when the user wishes to compile the project |
| Flow of Events | When the "compile" hyperlink is activated then the server will invoke the CAPS compiler (JGNAT). The server will return to the client a successful or not-successful dialog. If successful the project is bundled into an executable Java Archive File (JAR) file |
| Alternate Flow of Events | If service is unavailable at this time then a message to that affect will be delivered. |
| Pre-conditions | User logged on to the system. The project/version must be set by the user The server has all files available for the project in order to compile it to include the interface for the project. |
| Post-condition | User either has a successful or unsuccessful compilation. Either way a dialog is available for viewing. If successful then server creates an executable for the project |
| Special Requirements | N/A |

| ITEM | VALUE |
|---|---|
| Use Case Name | Executing a project |
| Brief Description | This use case describes the actions which will be performed by the application server when a user wishes to execute the prototype |
| Flow of Events | The user will execute the "execute" hyperlink which will in turn invoke the application server to execute the project and launch the executable JAR file in the client machine. |
| Alternate Flow of Events | N/A |
| Pre-conditions | User is logged on with the appropriate project/version set. User must have a successful compilation |
| Post-condition | The user is able to exercise his prototype on the local machine. |
| Special Requirements | N/A |

| ITEM | VALUE |
|---|---|
| Use Case Name | Managing files for the project |
| Brief Description | This use case describes the actions performed by the user in order to move files from the project folder to the client system and back again |
| Flow of Events | Using the CAPS Editor program the user will invoke the file manager. The user will then be shown the local project files as well as the project files on the server. The user will be able to download files to the client. Using an html interface the user will be able to select files to upload from the client machine to the project folder within the application server |
| Alternate Flow of Events | N/A |
| Pre-conditions | User is logged onto the system. |

| | A project/version has been set on the system. |
|---|---|
| Post-condition | Files either uploaded or downloaded successfully. |
| Special Requirements | N/A |

| ITEM | VALUE |
|---|---|
| Use Case Name | Communication b/n Application Server and DBMS |
| Brief Description | This use case describes the actions necessary to allow the application server to communicate with the database |
| Flow of Events | The application server will open a connection to the database.  If successful the application server will be able to update, insert, query, and view the data within the database. <br> If unsuccessful the application server will be notified |
| Alternate Flow of Events | If service is unavailable at this time then a message to that affect will be delivered. |
| Pre-conditions | Application server must have a medium with which to communicate with the database management system.  The database management system must allow for connection to the system as well as standard sql queries to be performed |
| Post-condition | Requested data is sent from the database to the application server in a format that is usable <br> Database is updated successfully if an update is requested |
| Special Requirements | N/A |

## 4. Sequence Diagrams

The following figures and the sequence diagrams developed from the use cases utilizing Rational Rose Enterprise 2001.
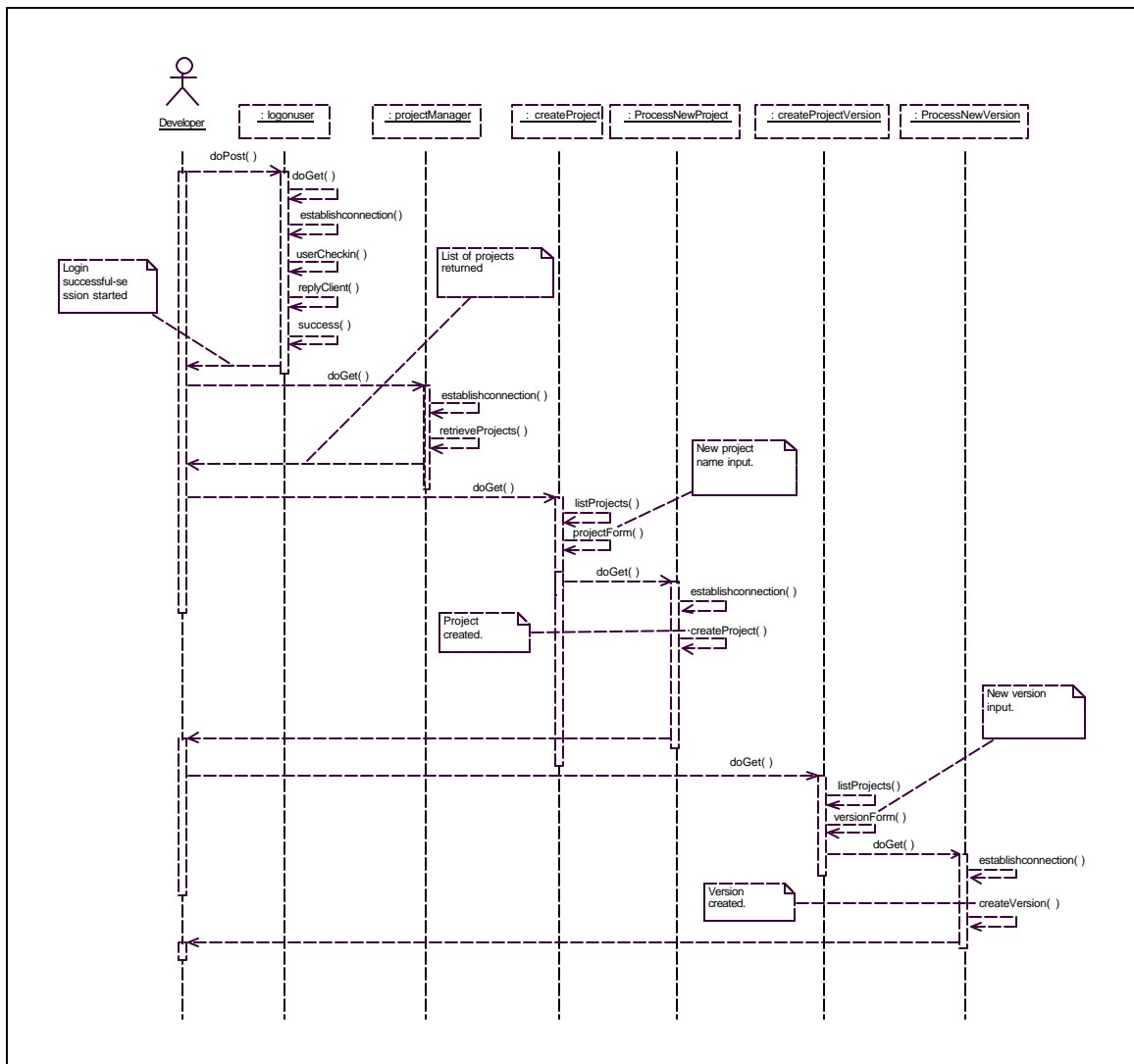
Figure 3.        Sequence Diagram User Login/Program Manager
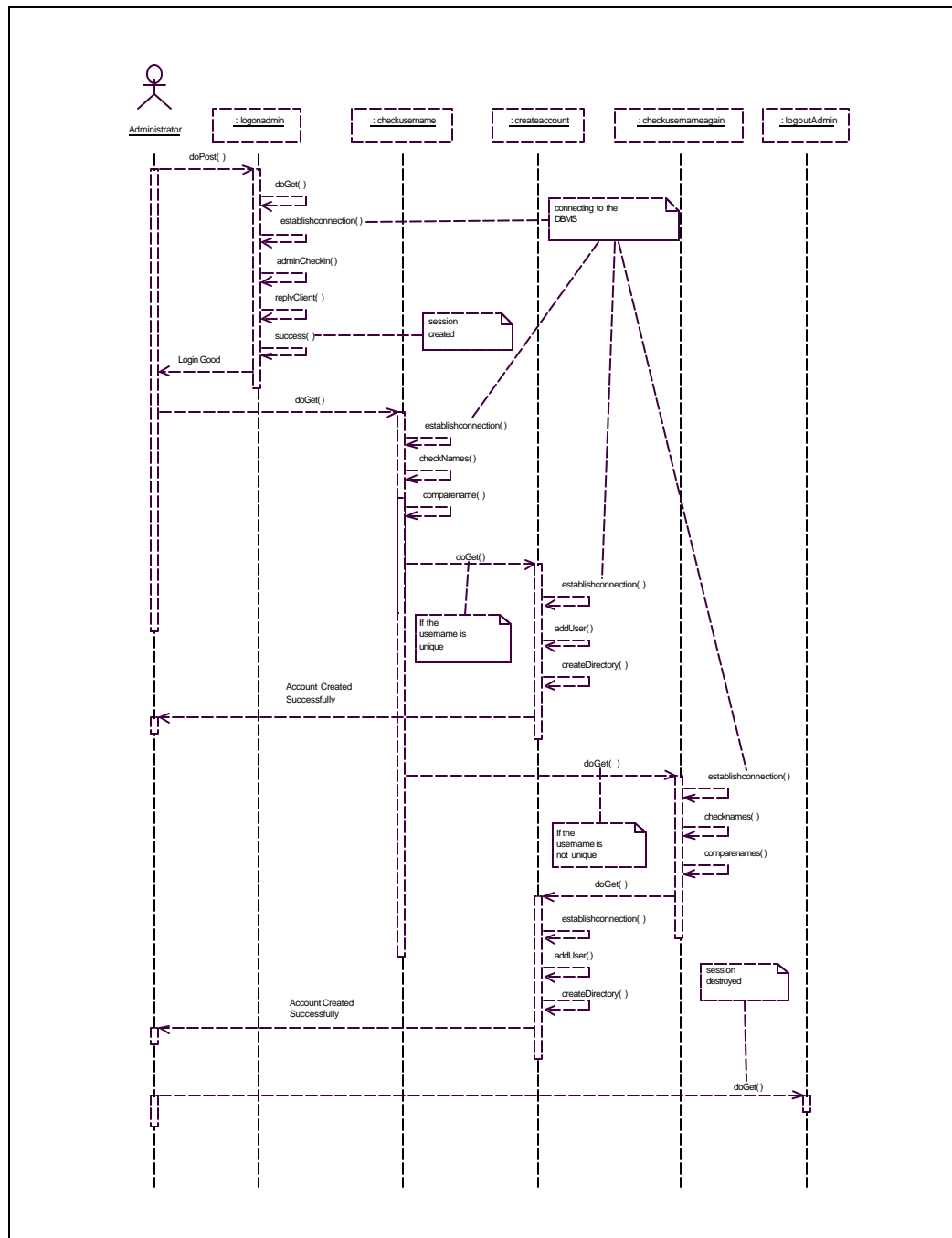
Figure 4.        Sequence Diagram Admin Account Creation
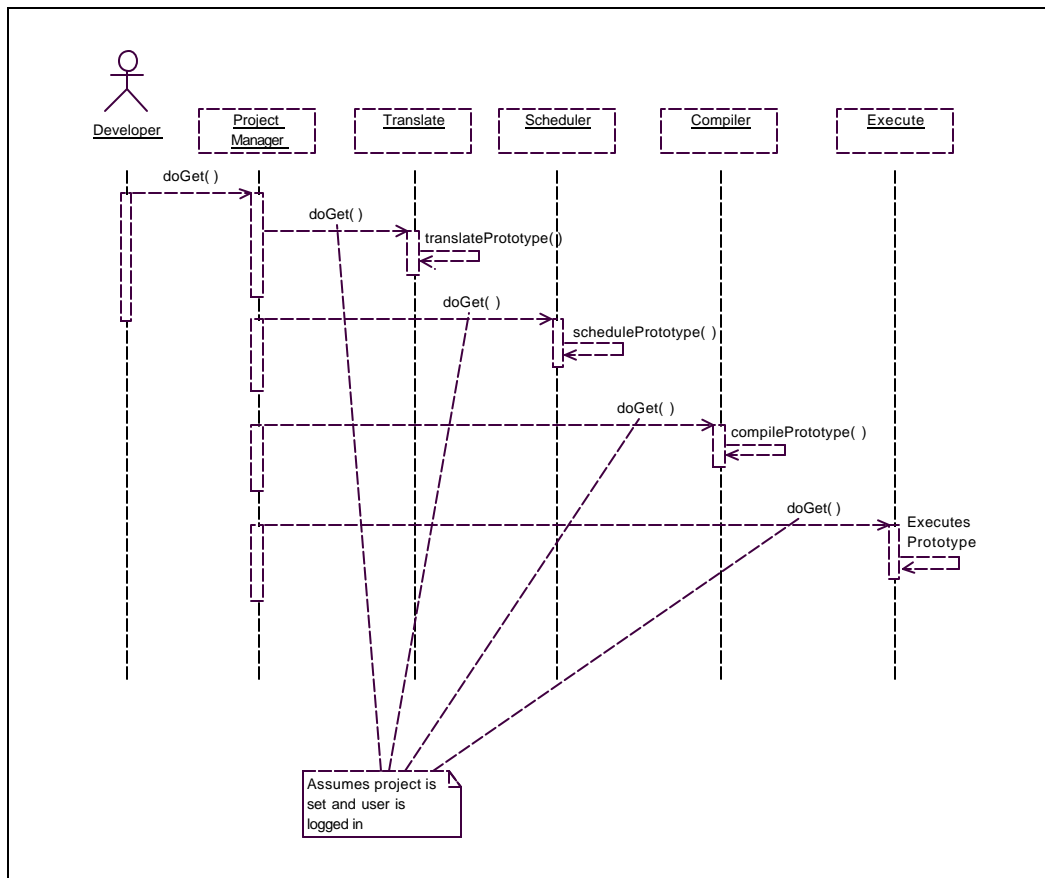
Figure 5.          Sequence Diagram Executive Support
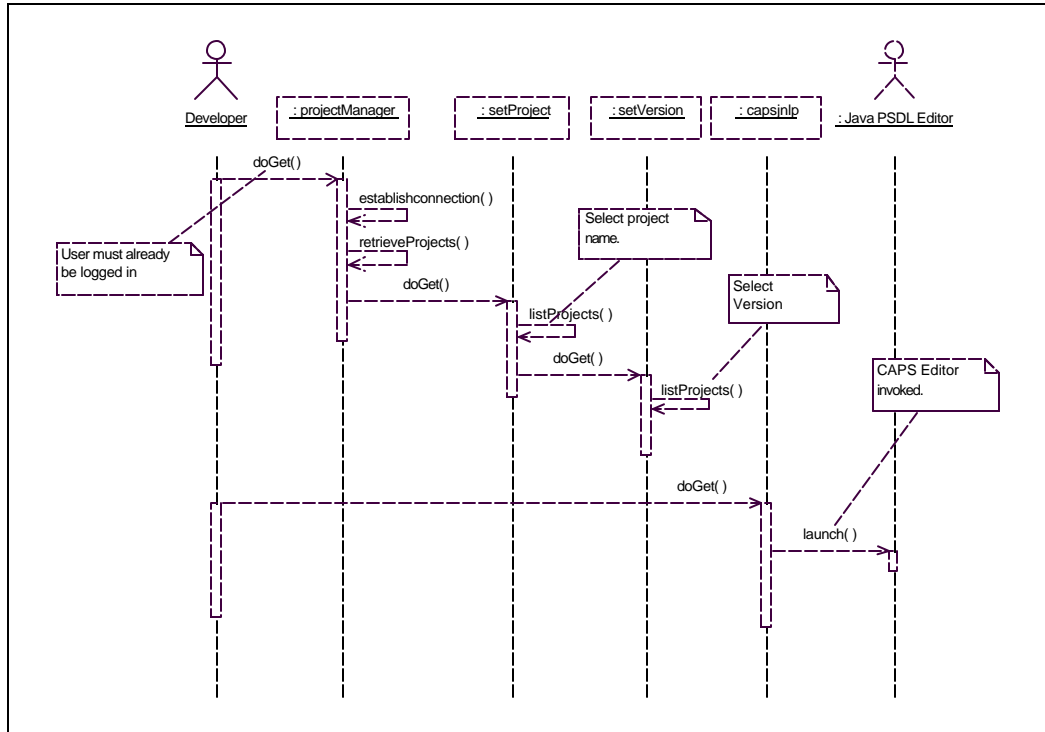
Figure 6.        Sequence Diagram Launch CAPS Editor

THIS PAGE INTENTIONALLY LEFT BLANK

## III.   WEB APPLICATION ARCHITECTURE SELECTION

As the Internet has continued to grow over the past
ten years a new type of application has entered the arena.
This application is known most commonly now as a web
application.  A web application is defined as a web system
(web server, network, HTTP, browser) where user input
(navigation and data input) affects the state of the
business.[7]  A web application is a software system
consisting of a front end interface delivered via the web
system, some middle system which implements the business
logic of the system, and a backend system consisting of a
database management system and/or a file system.  The
bottom line is that web applications implement business
logic.  This business logic in the case of SEAS is the
development of a prototype for an embedded real-time
system.  Most web applications of today are implemented
using a client server system.  More specifically they were
developed and implemented using a three tier distributed
client/server architecture.

### A.   THREE TIER ARCHITECTURE

The three tier client server architecture grew out of
the two tier client server architecture.  The two tier
architecture consisted of a user interface (client) and a
data management component (server). The two tier
architecture was designed to replace the file sharing
systems where files were downloaded to the client machine
manipulated and then uploaded back to the server.  The two
tier system allowed the client to connect directly to a
database management system, perform whatever process

23

necessary and then disconnect from the database management system.    This allowed for multiple users of the same database over a network.    Testing showed that the two tier architecture was limited to roughly 100 users.[14]    In an effort to improve upon this limitation a middle layer was added to the two tier system to provide process management where business logic and rules are executed and can accommodate hundreds of users.[14]    The three tier architecture is an effective, distributed client server design that provides increased performance, flexibility, maintainability, reusability, and scalability, while hiding the complexity of distributed processing from the user.[14]
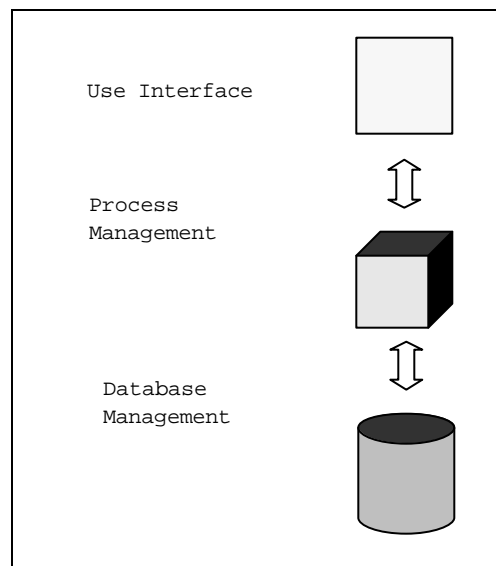


Figure 7.        Three Tier Architecture

The user interface tier in the web environment usually consists of a browser that handles session information, text input, dialog, and display management.    The middle tier provides process management services that are shared by multiple applications.    The main advantage to this is the centralization of process logic.    This makes

24

administration and change management more efficient and effective.  The third layer is the database management layer.  The third layer is dedicated to the management of data services as well as file services.  This ensures data consistency throughout the system.  It should be noted that the three tier architecture could be configured using multiple hardware configurations.  You could have one machine implement all three layers or add machines in any combination implementing one or more layers.  However, one advantage of the three tier architecture is that each layer can be built and implemented using different languages and different platforms.  This makes for a system that is flexible and scalable.

## B.  PROPOSED ARCHITECTURE FOR SEAS

At the outset of the development of SEAS as a web application the necessary sub-systems needed to execute SEAS were available.  This included the translator, scheduler, and compiler of the executive support system as well as the PSDL editor implemented in the Java programming language.  Although by itself the PSDL editor would only allow a user to produce the PSDL code for a prototype, it was clear that it could be modified to work as a user interface for a web based SEAS system giving the user a powerful client side tool.  Another benefit of the Java based interface is the portability across platforms.  This would meet a very important requirement of system portability.  The fact that these components were already developed and separate allowed for an easy mapping to a three tier web based client server application.  The system could use the PSDL Editor and browser for the user

interface system, an application server could add the
functionality of business logic as well as a platform to
run the execution support needed (i.e.: translation,
scheduling, and compiling), and a database management
system/file system could provide the information and
persistent storage necessary to implement the system.
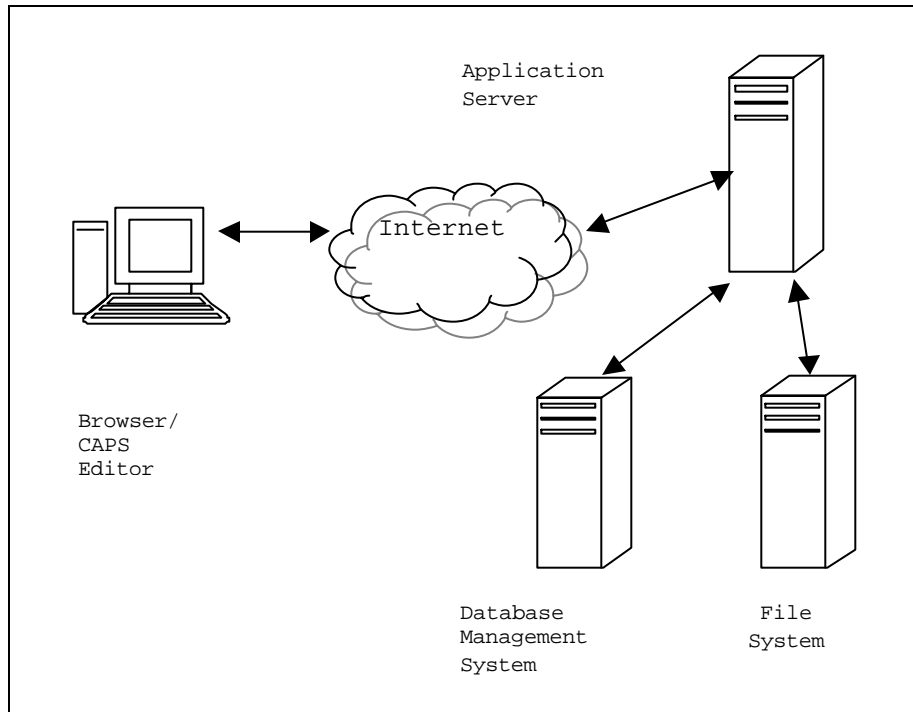


Figure 8.          SEAS Web Architecture


## C.   SUPPORTING TECHNOLOGY

Web applications currently deployed in business today
depend on a variety of supporting software to enable them
to function.  In the early 1990's one computer software
language was developed with the premise that it would be
able to be used heavily for web applications.  The language
was Java.  Since its inception the developers of Java have
developed application program interfaces (API) to help
software engineers develop applications.  API's are a set

of tools provided to the developer to accomplish certain task.  We found that many of the API's currently developed would be able to be utilized in the development of SEAS as a web application.  Those API's included Java Servlet 2.2, Java Web Start 1.0.1, and Java Database Connectivity (JDBC) 2.0.

## 1.  Java Servlet 2.2

Java Servlet technology provides web developers with a simple, consistent mechanism for extending the functionality of a web server and for accessing existing business systems. Servlets provide a component-based, platform-independent method for building web-based applications, without the performance limitations of CGI programs. Servlets are server and platform independent. Servlets have access to the entire family of Java APIs, including the JDBC API to access databases. Servlets can also access a library of HTTP-specific calls and receive all the benefits of the mature Java language, including portability, performance, reusability, and crash protection.  Today, servlets are a popular choice for building interactive web applications.[11]  The greatest capability provided by Java Servlets is the fact that the entire family of Java API's are available to the programmer.  Servlets receive information from the client's browser and performs the requested action on that information.  In this application the Java Servlets will receive information from the user and dynamically create the HTML page necessary for the response.

## 2. Java Web Start 1.0.1

Java Web Start would be utilized in this system as a means of invoking the PSDL Editor as well as the execution of the developed prototype. Java Web Start gives the user the power to launch full-featured applications from the Internet. Java Web Start alleviates the user from performing complicated installation procedures. Java Web Start includes the security features of the Java 2 platform, so the integrity of your data and files is never compromised. In addition, Java Web Start technology enables the user to use the latest Java 2 technology with any browser. The user launches applications simply by clicking on a Web page link. If the application is not present on the user's computer, Java Web Start automatically downloads all necessary files. It then caches the files on the user's computer so the application is always ready to be re-launched anytime the user wants either from an icon on the user's desktop or from the browser link. And no matter which method the user chooses to launch the application, the most current version of the application is always presented. Java Web Start is easy to use: the user can efficiently access, download, and launch applications as easily as accessing a Web page. Because Java Web Start runs independently of any browser, the user can shut down the browser or surf to another Web page and continue using the application. Applications deployed with Java Web Start do not require a browser interface to operate. To save time later, the user can re-launch the desired application anytime from the desktop icon, without powering up the browser again.[12] Java Web Start's flexibility and functionality allowed us to develop the user interface

28

efficiently and effectively. Java Web Start provided us with the mechanisms necessary to manage the change during the development of the application. This capability was crucial in our development.

### 3. JDBC Data Access API

JDBC technology is an API that lets the user access virtually any tabular data source from the Java programming language. It provides cross-DBMS connectivity to a wide range of SQL databases. The JDBC API allows developers to take advantage of the Java platform's "Write Once, Run Anywhere" capabilities for industrial strength, cross-platform applications that require access to enterprise data. With a JDBC technology-enabled driver, a developer can easily connect all corporate data even in a heterogeneous environment.[16] Although the database needed for implementing our system was not complicated, JDBC gave us the freedom to separate the DBMS component from the application server allowing us to take full advantage of the three tier architecture of web applications.

29

THIS PAGE INTENTIONALLY LEFT BLANK

# IV. IMPLEMENTATION

Once the architecture was chosen for the SEAS system the need to develop the components necessary to fill that architecture was the next step. Today the marketplace is filled with application servers, database management systems (DBMS) as well as hardware platforms to implement the layers of the systems. Maintaining flexibility of the overall system as well as fiscal constraints drove the decisions made in choosing the appropriate systems. Flexibility would be maintained by the use of the Java Application Programming Interfaces (API) mentioned in the previous chapter. Fiscal constraints would be met by the use of proven open source systems to fulfill the application server as well as the DBMS.

## A. USER INTERFACE

The user interface would be broken down into two separate systems. The first system would be a standard Internet browser, preferably Internet Explorer 5.0 or Netscape Navigator 4.X. The browsers would allow for interaction between the client and the server. Through the use of HTML forms and hyperlinks the user would be able to navigate and utilize the system. The second system would be a modified version of the PSDL Editor. The latest version was developed in September 2000 by Shen-Yi Tao.[6] The PSDL Editor was originally designed as a standalone system for the implementation of SEAS. The problem quickly arose as to how to integrate the PSDL Editor into SEAS without degrading the benefits of maintainability and flexibility of the three tier web architecture. The use of

31

Java Web Start to implement the execution of the PSDL Editor was chosen. The PSDL Editor was already developed sufficiently enough to be used by SEAS. Java Web Start would allow us to reuse this component with little to no change necessary. The Java Community Process has developed Java Web Start over the past year. The technology underlying Java Web Start is the Java Network Launching Protocol and API (JNLP). Java Web Start is the reference implementation for the JNLP specification. The JNLP technology defines, among other things, a standard file format that describes how to launch an application called a JNLP file. Once the JNLP file is created it is placed into a web page as a hyperlink. When the hyperlink is activated the server activates Java Web Start on the client machine and the application is launched. For us to utilize the PSDL Editor we would have to have a means of synchronizing the PSDL Editor to the application server to include the user, prototype name, and prototype version. The JNLP file format allows a user to specify specific parameters to be used by the application upon its invocation. We needed to be able to add these specific parameters for each individual user when the user launches the PSDL Editor. We were able to accomplish this task by dynamically building the JNLP file for the user to launch the PSDL Editor with the specific parameters needed for a single prototype. The dynamic building of the JNLP file was accomplished through the use of Java Servlet Technology. An example of the dynamically created JNLP file is listed below. Highlighted are the parameters dynamically added by the Java servlet controlling the creation of the JNLP file used in launching the PSDL Editor.

32

```
<?xml version="+"\"1.0\" encoding="+"\"utf-8\"?>
<!-- JNLP File for CAPS Application -->
<jnlp");
        spec="1.0+"
        codebase=""http://131.120.9.87:8080/CapsWeb>
        codebase="http://seaotter.cs.nps.navy.mil:8080/CapsWeb">

        <information>
                <title>Distributed CAPS Application</title>
                <vendor>Software Engieering Group, Naval Postgraduate
                School</vendor>
                <homepage href="WebCaps.htm"/>
                <description>PSDL Distributed CAPS Application</description>
                <description kind="short">The PSDL Editor for the Distributed
                CAPS Application.</description>
                <icon href="globi.gif"/>
                <offline-allowed/>
        </information>
        <security>
        <all-permissions/>
        </security>
        <resources>
                <j2se version="1.3"/>
                <jar href="testCaps_6.jar"   download="eager"/>
        </resources>
    <application-desc   >
    <argument>userUserName</argument>
    <argument>selectedVersion</argument>
    <application-desc/>

</jnlp>
```

Figure 9.        JNLP File Example

## 1.   Browser Interface

By utilizing a web browser and HTML we were able to
implement an interface to the SEAS system that was simple,
efficient, and effective.  Although the application server
utilized Java servlets in its implementation of the
business logic of the system the output was still in the

33

form of an HTML web page.  Except for three static web pages all other pages were constructed based on the information provided to the application server by the user or by other servlets performing in the application server. Being able to output information to the user dynamically allows for a fine control as to what information the user is provided.  Our design focused no providing the user the information the user needed as to what the status of the development of the prototype.  An important part of the interface was the use of error logs to display to the user problems encountered during translation, scheduling, and compiling of the prototype.  If an error was encountered it would be output to the error logs that was then available to the user for viewing.  Future enhancements to this interface can be easily implemented with the addition of Java Servlets to produce the desired output to the HTML web page.

### 2.  Modification to PSDL Editor

The PSDL Editor, implemented in Java, allowed a user to create the PSDL model of a prototype system.  It would create the PSDL file as well as automatically generate the target programming language source code templates necessary for all operators created for the prototype.  The PSDL Editor would need to be modified to meet the requirements necessary to implement the SEAS system on the Internet. Many of the modification allowed for the use of Java Web Start described above.  These modifications included the addition of input parameters of user name, project name and project version.  This modification would be needed for the instantiation of the PSDL Editor using Java Web Start.  A specific project set via the logic in the application

server would ensure that the PSDL Editor was synchronized to work with the same project as was being used by the user on the application server. A second modification was the addition of a file manager. The file manager would allow the user to transfer prototype specific files to and from the file management system on the application server. This would allow the user to utilize different client systems as well as provide the files necessary for the application server to translate, schedule, compile and execute the prototype. The third modification to the PSDL Editor was the addition of a text editor allowing the user to edit Ada template files generated by the PSDL Editor in order to implement logic specific to the prototype under development. The final modification to the PSDL Editor was a function to allow the user to add support files to the project. These support files contain the source code and/or binary objects of the user-supplied reusable components for implementing the prototype. The project file includer would allow the user to specify a file and have it automatically added to the prototype folder. The user could then upload the file to the server where it would be included in the compilation of the project and inevitably needed for the execution of the prototype.

### 3. PSDL Editor UML Modeling

The following UML use case diagram models the modifications and functions specific to the PSDL Editor.

The CAPS Editor is Launched with the username, projectname and project version set by the application server

Developer

PSDL Editor

Local File System

Project File Includer

File Manager

Application_Server    File_System

Downloads files necessary for project

Ada Code Editor
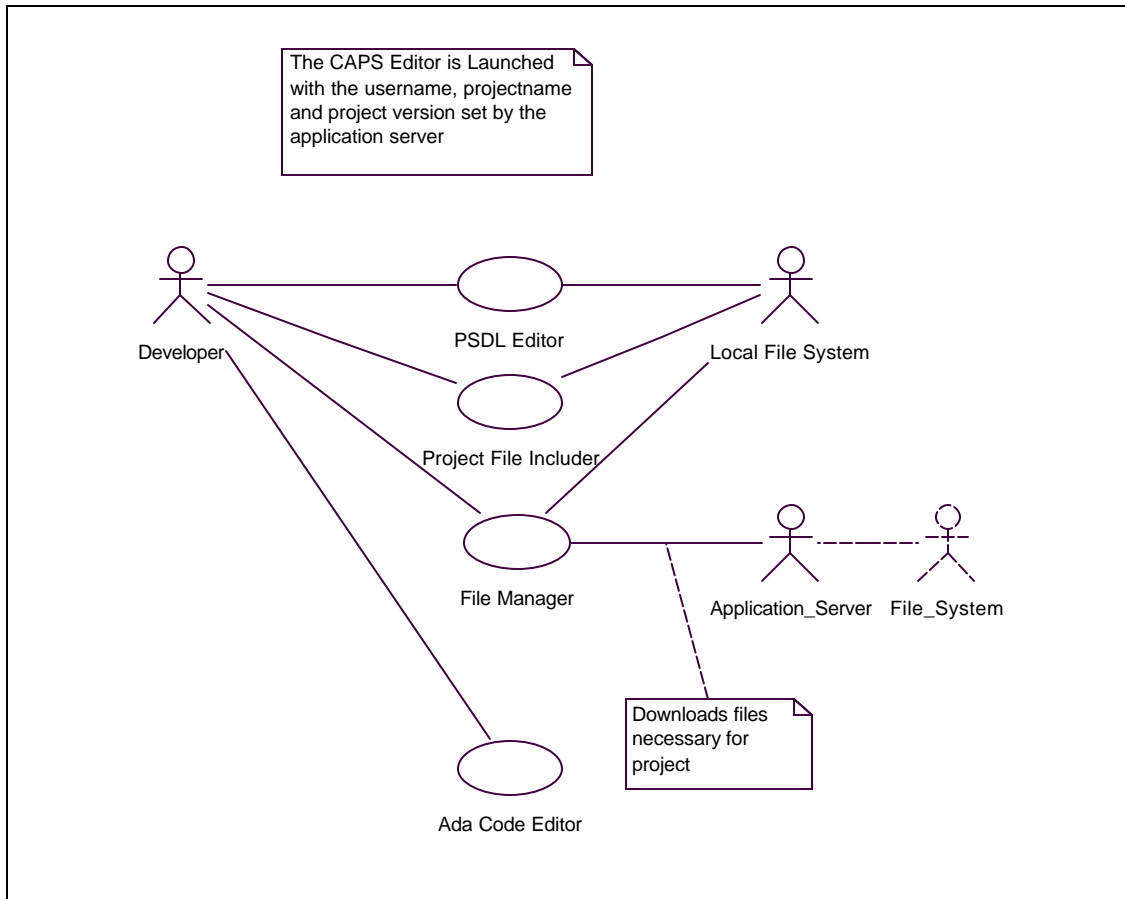
Figure 10.        PSDL Editor Use Case Diagram

When the user launches the PSDL Editor he/she is now able to develop the prototype on the local client system. If the project is new the user can open the PSDL Editor function and begin to model the prototype.  If it is an existing project the user can invoke the file manager and download project files to the local system and then begin to work on the prototype.  Once complete the user is then able, via a servlet, to upload the project files back to the server for further development.

**B.   APPLICATION SERVER**

The middle layer of the SEAS web application would be the control center of the system.  It would have to process the logic of the system concerning access, creation of

36

projects as well as development and execution of the project. In order to perform these functions as well as maintain portability of this portion of the system the decision was made to choose an application server which implements Sun Microsystems' Java Servlet API. The application server chosen to implement this API was Tomcat-Jakarta 3.2.3. Tomcat is an open source application server developed under the Apache project. There is a version capable of running on Windows, UNIX, and Linux. Therefore servlets written for the server running on a Windows machine could be loaded on a UNIX machine running the same Tomcat server without having to recompile the code. The Tomcat application server has the ability to serve both static and dynamic web pages. This precluded the need for a web server, although a web server inclusion in future upgrades could be used to optimize the system.

**1.  Java Servlet Implementation for the User System**

The business logic of the SEAS system was implemented using Java Servlets. The servlets' functions mapped to the use cases developed during the design phase of the development. The major requirements needed for the system would be implemented by the servlets. One group of servlets would deal with user and administrator access. A second group would deal with launching the PSDL Editor as well as uploading and downloading files to the server. Another group of servlets would implement the project management phase that would allow for the translation, scheduling, compilation, and execution of a project. The state diagram in figure 3 depicts the flow of the business logic of the system. Once the application server is started it is initialized and ready to receive requests

from user.   A user logs into the system with a user name
and password that is passed to a login servlet via an HTML
form.    The  login  servlet  connects  to  the  database  and
determines if the user is in fact an authorized user.   If
the user is authorized to use the SEAS system, the *"login"*
servlet generates a response to the user via HTML and shows
the user that he/she is logged into the system and presents
the user with the first of two project managers.   The first
project  manager  allows  the  user  to  create  projects.   Upon
creating a project the user supplies the *"project creation"*
servlet  with  a  project  name  and  version.    The  *"project
creation"* servlet checks to ensure the project is unique
and  if  it  is  adds  the  project  to  the  database  and  also
creates  the  workspace  folders  in  the  file  system  for  the
project.    The  user  is  returned  to  the  first  project
manager.   The user would then identify the project he/she
would  like  to  work  on.    The  user  is  able  to  select  a
project  from  an  HTML  option  form.    The  project  name  and
version is sent to the *"setProject"* servlet which in turn
sets  the  project  name  and  version  in  the  system  and  now
allows  the  user  to  do  the  following:    launch  the  PSDL
Editor,  translate,  schedule,  compile,  or  execute  the
project.   Each of the above capabilities is handled by a
servlet that in turn invokes the proper action as requested
by the user.   The user is able to log out of the system at
anytime after he/she has logged in.

**2.  Java Servlet UML Modeling for the User**

The  following state diagram models the behavior of the
servlets utilized to implement the SEAS system on the web.

38

Figure 11.        Servlet State Diagram
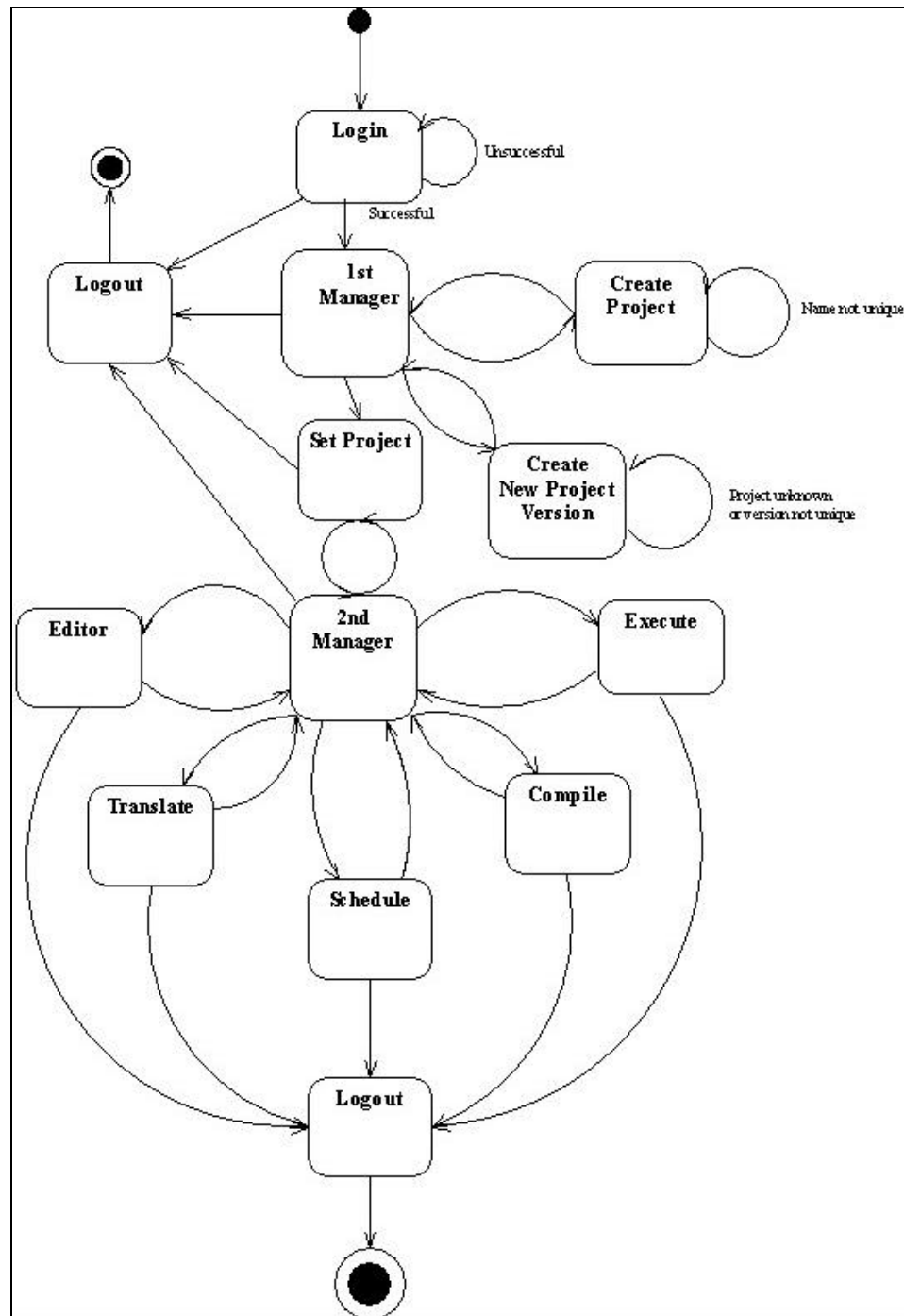
## C.    THE DATABASE MANAGEMENT SYSTEM

The decision to keep information concerning projects and users in a database was made after the initial design

requirements were completed.  A simple relational database management system (DBMS) would be needed to handle the requirements of the system.  The decision was made to utilize an open source relational database management system called MySQL.  MySQL is a reliable relational DBMS with versions developed for Windows, UNIX and Linux systems.  MySQL has a JDBC:ODBC bridge that is network capable allowing for the database management system to be deployed, if needed, on a separate system from the application server.  This bridge allows servlets running on the application server to utilize Java Database Connectivity (JDBC) to query and update the database deployed for SEAS.  The database would store personal data for each user and project information pertaining to each user project developed.  In an effort to increase performance the database management system was implemented on a separate machine from the application server.

**1.  Entity Relationship Diagram**

The following entity relationship diagram is shown below to illustrate the database implemented in the SEAS system.
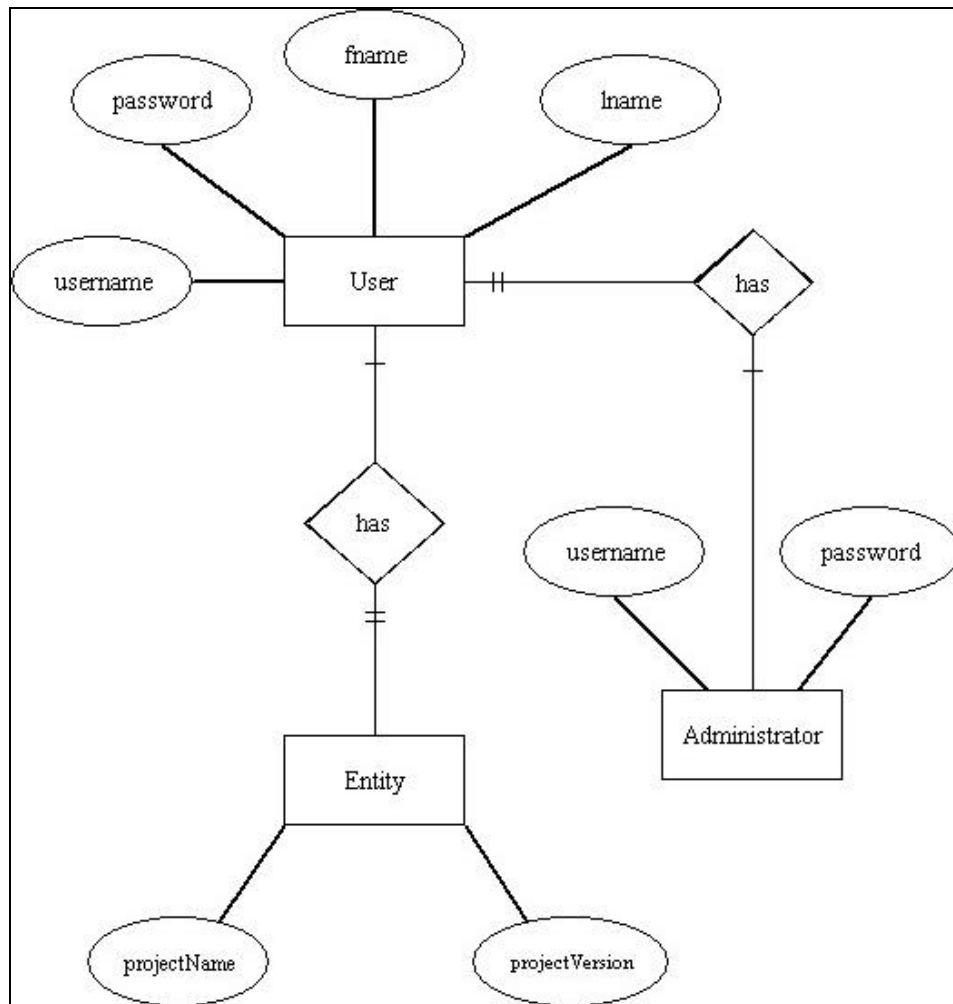
Figure 12.        ER Diagram for SEAS DBMS

## D.   EXECUTING THE PROTOTYPE

Once the user has translated and scheduled the prototype he/she is ready to add the files and logic necessary to finish and execute the prototype.  The user would now take the Ada template files generated by the system and add the logic necessary for proper execution. After that is completed a graphical user interface would be developed and added to the prototype.  These files would then be added to the project via the *"upload"* servlet. This servlet allows the user to choose a series of files to

41

upload to the server for inclusion in the development of the prototype. The user would then invoke the *"compiler"* hyperlink that in turn would activate the *"compiler"* servlet. This servlet would invoke the compiler on the designated project. The compiler uses GNATchop to separate the Ada files and then if successful uses JGNAT to compile the project into an executable Java Archive File (JAR). Once complete the user would activate the *"execute"* hyperlink that in turn would invoke the *"execute"* servlet. The *"execute"* servlet creates a JNLP file dynamically, adding the parameters necessary, launching the prototype on the client system via Java Web Start.

# V.    CONCLUSIONS AND RECOMMENDATIONS

## A.    CONCLUSIONS

The development of the SEAS system could not have come at a better time.   As the system was being designed an implemented it was being utilized by students.   As new functionality was added to the system the users were able to utilize it.   This capability to use the system seamlessly during the constant changes was due to the three tier architecture and the products and implementation languages chosen for the system.   As the PSDL Editor was changed and functionality added that change was managed by Java Web Start.   When initiated the Java Web Start would check for an updated version of the software.   If available it would download and launch the new system.   The use of Java Servlets and an application server capable of running on multiple platforms allowed for changes in the system topology as well as system upgrades.   In the end the system had accomplished the same functionality as the original CAPS system implemented on the UNIX system.

## B.    RECOMMENDATIONS

Now that the initial working implementation of a web based SEAS system has been used it should be noted that improvements should be identified and implemented in the system.   The following are a series of initial recommendations.

### 1.    PSDL Editor Upgrades

At this time the basic functionality needed by the CAPS Editor has been implemented.   It would be a good idea now to step back and look at how the Editor is implemented

43

and determine if any additional functionality, if added, would improve the overall system.  As an example, the file manager could be expanded to allow for not only the downloading of files but also for the uploading of files. An overall usability study would greatly benefit the system and its users.

### 2.  Optimization of Servlet Code

The final recommendation for the initial implementation of a web based SEAS system is the optimization of the servlet code developed for the business logic of the system.  One major optimization would be the addition of "connection pool" for the database management system.  This is not needed when the usage is low but as the system becomes more popular and its usage increases it is safe to say that the ability to eliminate potential bottlenecks when connecting to the database would be crucial.  "Connection pooling" is a means of initializing a maximum number of connections to the database, each in a separate thread.  The pooling mechanism would manage when a connection was needed and if available allocate the connection to the user.

# APPENDIX A SERVLET CODE

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;

/**
 * This class dynamically creates a .jnlp file used to launch the CAPS Editor
 * @see java.servlet.http
 * @author J. A. McDonald III
 */
public class capsjnlp extends HttpServlet {

    String selectedProject = "";
    String selectedVersion = "";
    String userUserName = "";

    private PrintWriter  out;

    HttpSession session;

    HttpServletRequest request;
    HttpServletResponse response;


/**
* Processes the user request and responds via an HTML output
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
*/
 public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException{

    res.setContentType("application/x-java-jnlp-file");

    request = req;
    response = res;
    session=request.getSession();

    selectedProject = (String)session.getAttribute("selectedProject");
    selectedVersion = (String)session.getAttribute("selectedVersion");
    userUserName = (String)session.getAttribute("username");

    out = response.getWriter();

    out.println("<?xml version="+"\"1.0\" encoding="+"\"utf-8\"?>");
    out.println("<!-- JNLP File for CAPS Application -->");
    out.println("<jnlp");
    out.println("    spec="+"\"1.0+\"");
    //out.println("   codebase="+"\"http://131.120.9.87:8080/CapsWeb\">");
    out.println("    codebase="+"\"http://seaotter.cs.nps.navy.mil:8080/CapsWeb\">");

    out.println("    <information>");
    out.println("            <title>Distributed CAPS Application</title>");
    out.println("            <vendor>Software Engieering Group, Naval Postgraduate
School</vendor>");
    out.println("            <homepage href="+"\"WebCaps.htm\"/>");
    out.println("            <description>PSDL Distributed CAPS
Application</description>");
    out.println("            <description kind="+
    "\"short\">The PSDL Editor for the Distributed CAPS Application.</description>");
    out.println("                <icon href="+"\"globi.gif\"/>");
    out.println("            <offline-allowed/>");
    out.println("    </information>");
```

45

```
       out.println("    <security>");
       out.println("          <all-permissions/>");
       out.println("    </security>");
       out.println("    <resources>");
       out.println("             <j2se version="+"\"1.3\"/>");
       out.println("             <jar href="+"\"testCaps_6.jar\"
download="+"\"eager\"/>");
       out.println("    </resources>");
       out.println("<application-desc   >");
       out.println("<argument>"+userUserName+"</argument>");
       out.println("<argument>"+selectedProject+"</argument>");
       out.println("<argument>"+selectedVersion+"</argument>");

       out.println("    <application-desc/>");

       out.println("</jnlp>");

       out.close();

    }


 }
```

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;


/**
 * This class checks the usernames in the database for uniqness
 *
 * @see javax.servlet.http
 * @author J. A. McDonald
 */
public class checkusername extends HttpServlet {

    private String userFirstName;
    private String userLastName;
    private String userUserName;
    private String userPassWord;

    private String driverName = "org.gjt.mm.mysql.Driver";
    private String urlName = "jdbc:mysql://131.120.9.87:3306/capsdata";

    private Driver       driver;

    private Connection   connection;

    private PrintWriter  out;

    private RequestDispatcher rd;

    HttpSession session;

    HttpServletRequest request;
    HttpServletResponse response;
/**
* Processes the user request and responds via an HTML output
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
*/
public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException{

    res.setContentType("text/html");

    request = req;
    response = res;

    session = req.getSession();

        //Get the info
    userFirstName = req.getParameter("fname");
    userLastName = req.getParameter("lname");
    userUserName = req.getParameter("username");
    userPassWord = req.getParameter("password");

    session.setAttribute("userfirstname", userFirstName );
    session.setAttribute("userlastname", userLastName );
    session.setAttribute("userpassword", userPassWord );

    // Gets the PrintWriter object for sending HTML commands
    out = res.getWriter();

    if(!checknames()){

        topOfPage();
        out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
        "\"#0000FF\"><b><i>Please select another username for the
account.</i></b></font></p>");
```

47

```
        out.println("<center><form method ="+"\"get\" action ="+
        "\"/CapsAdmin/servlet/checkusernameagain\"><font color=blue> Username:<input
type=text size="+"\"12\" name ="+
        "\"username\">");

        out.println("<input type="+"\"submit\" value="+
        "\"Submit\">  <input type=reset></form></center>");

        bottomOfPage();

    }
    else{

        session.setAttribute("userusername", userUserName );
        topOfPage();
        sendtocreate( req, res );
        out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
        "\"#0000FF\"><b><i>Account Created Successfully!</i></b></font></p>");
        bottomOfPage();


    }

}//end of doGet

/**
* Establishes a connection to the database
*/
private void establishConnection( ){

    Properties props = new Properties();

    props.put("user",     "mcdonald");
    props.put("password", "mcdonald");

    try {

        driver   = (Driver) Class.forName( driverName ).newInstance();
        connection = driver.connect( urlName, props );

    }
    catch ( Exception e ){

        out.println("<p> Error: Cannot establish a connection to the database");

    }

}//end of establishConnection

/**
*  @return a <code>boolean</code>
*/
private boolean checknames(){

    boolean usernamestatus = false;

    establishConnection();

    Statement sqlStmt = null;
    ResultSet rs      = null;

    String name = "";

    try {

        String query = "select username from user";

        sqlStmt = connection.createStatement( );
```

48

```java
        rs      = sqlStmt.executeQuery( query );

        Vector names = new Vector();

        while ( rs.next() ){

            name = rs.getString("username");

            names.add(name);


        }

    //close connection to the database
    connection.close();
    sqlStmt.close();

    usernamestatus = comparename( userUserName, names );


}
catch (Exception e){

    out.println(e);

}

return usernamestatus;



}//end of checknames

/**
* Outputs the top of the HTML Page
*/
private void topOfPage(){

    out.println("<html><head><title>Distributed Computer Aided Prototyping
System</title>");
    out.println("<style fprolloverstyle>A:hover {color: #FF0000; font-size: 14pt;
vertical-align: 10; font-weight: bold}</style>");
    out.println("</head><hr color="+"\"#0000FF\">");

    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
        "\"#0000FF\"><b><i>Distributed Computer Aided Prototyping
System</i></b></font></p>");

    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
        "\"#0000FF\"><b><i>Administrator Tools</i></b></font></p>");

    out.println("<hr color="+"\"#0000FF\"><br>");

}//end of topOfPage

/**
* Outputs the bottom of the HTML Page
*/
private void bottomOfPage(){

    out.println("<br><br><br>"+"<hr color="+"\"#0000FF\">");

    out.println("<center><a href="+"\"/CapsAdmin/createaccountattemp.html\">Create
Another Account</a></center>");

    out.println("<center><a
href="+"\"/CapsAdmin/servlet/logoutAdmin\">Logout</a></center>");
```

```java
        out.println("</html>");


    }//end of bottomOfPage

    /**
    * @return a <code>boolean</code>
     *@param name used to compare names in database
     *@param in Vector of usernames in database
    */
    private boolean comparename(String name, Vector in){

        String temp = "";

        Enumeration e = in.elements();

        while( e.hasMoreElements() ){

            temp = (String)e.nextElement();

            if( name.equals(temp) ){

                return false;

            }

        }

        return true;

    }//end of comparname

    /**
    * Sends request to create the account
     *@param HttpServletRequest req
     *@param HttpServletResponse res
     *@throws IOExecption
    */
    private void sendtocreate(HttpServletRequest req, HttpServletResponse res)throws
    IOException{

        rd = getServletContext().getRequestDispatcher("/servlet/createaccount");

        try{

            rd.include( req, res );

        }
        catch( IOException e ){

            out.println("ProblemIO");

        }
        catch( ServletException e ){

            out.println("ProblemServlet");
            out.println(e);

        }

    }//end of sendtocreate


}//end of checkusername
```

50

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;


/**
 * This class checks the usernames in the database for uniquness
 *
 * @see javax.servlet.http
 * @author J. A. McDonald
 */
public class checkusername extends HttpServlet {

    private String userFirstName;
    private String userLastName;
    private String userUserName;
    private String userPassWord;

    private String driverName = "org.gjt.mm.mysql.Driver";
    private String urlName = "jdbc:mysql://131.120.9.87:3306/capsdata";

    private Driver       driver;

    private Connection   connection;

    private PrintWriter  out;

    private RequestDispatcher rd;

    HttpSession session;

    HttpServletRequest request;
    HttpServletResponse response;
/**
* Processes the user request and responds via an HTML output
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
*/
public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException{

    res.setContentType("text/html");

    request = req;
    response = res;

    session = req.getSession();

        //Get the info
    userFirstName = req.getParameter("fname");
    userLastName = req.getParameter("lname");
    userUserName = req.getParameter("username");
    userPassWord = req.getParameter("password");

    session.setAttribute("userfirstname", userFirstName );
    session.setAttribute("userlastname", userLastName );
    session.setAttribute("userpassword", userPassWord );

    // Gets the PrintWriter object for sending HTML commands
    out = res.getWriter();

    if(!checknames()){

        topOfPage();
        out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
        "\"#0000FF\"><b><i>Please select another username for the
account.</i></b></font></p>");
```

51

```java
        out.println("<center><form method ="+"\"get\" action ="+
        "\"/CapsAdmin/servlet/checkusernameagain\"><font color=blue> Username:<input
type=text size="+"\"12\" name ="+
        "\"username\">");

        out.println("<input type="+"\"submit\" value="+
        "\"Submit\">  <input type=reset></form></center>");

        bottomOfPage();

   }
   else{

        session.setAttribute("userusername", userUserName );
        topOfPage();
        sendtocreate( req, res );
        out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
        "\"#0000FF\"><b><i>Account Created Successfully!</i></b></font></p>");
        bottomOfPage();


   }

}//end of doGet

/**
* Establishes a connection to the database
*/
private void establishConnection( ){

    Properties props = new Properties();

    props.put("user",     "mcdonald");
    props.put("password", "mcdonald");

    try {

        driver   = (Driver) Class.forName( driverName ).newInstance();
        connection = driver.connect( urlName, props );

    }
    catch ( Exception e ){

        out.println("<p> Error: Cannot establish a connection to the database");

    }

}//end of establishConnection

/**
*  @return a <code>boolean</code>
*/
private boolean checknames(){

    boolean usernamestatus = false;

    establishConnection();

    Statement sqlStmt = null;
    ResultSet rs      = null;

    String name = "";

    try {

        String query = "select username from user";

        sqlStmt = connection.createStatement( );
```

52

```java
            rs      = sqlStmt.executeQuery( query );

            Vector names = new Vector();

            while ( rs.next() ){

                name = rs.getString("username");

                names.add(name);


            }

        //close connection to the database
        connection.close();
        sqlStmt.close();

        usernamestatus = comparename( userUserName, names );


    }
    catch (Exception e){

        out.println(e);

    }

    return usernamestatus;



}//end of checknames

/**
* Outputs the top of the HTML Page
*/
private void topOfPage(){

    out.println("<html><head><title>Distributed Computer Aided Prototyping
System</title>");
    out.println("<style fprolloverstyle>A:hover {color: #FF0000; font-size: 14pt;
vertical-align: 10; font-weight: bold}</style>");
    out.println("</head><hr color="+"\"#0000FF\">");

    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
        "\"#0000FF\"><b><i>Distributed Computer Aided Prototyping
System</i></b></font></p>");

    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
         "\"#0000FF\"><b><i>Administrator Tools</i></b></font></p>");

    out.println("<hr color="+"\"#0000FF\"><br>");

}//end of topOfPage

/**
* Outputs the bottom of the HTML Page
*/
private void bottomOfPage(){

    out.println("<br><br><br>"+"<hr color="+"\"#0000FF\">");

    out.println("<center><a href="+"\"/CapsAdmin/createaccountattemp.html\">Create
Another Account</a></center>");

    out.println("<center><a
href="+"\"/CapsAdmin/servlet/logoutAdmin\">Logout</a></center>");
```

```java
    out.println("</html>");


}//end of bottomOfPage

/**
* @return a <code>boolean</code>
 *@param name used to compare names in database
 *@param in Vector of usernames in database
*/
private boolean comparename(String name, Vector in){

    String temp = "";

    Enumeration e = in.elements();

    while( e.hasMoreElements() ){

        temp = (String)e.nextElement();

        if( name.equals(temp) ){

            return false;

        }

    }

    return true;

}//end of comparname

/**
* Sends request to create the account
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOExecption
*/
private void sendtocreate(HttpServletRequest req, HttpServletResponse res)throws
IOException{

    rd = getServletContext().getRequestDispatcher("/servlet/createaccount");

    try{

        rd.include( req, res );

    }
    catch( IOException e ){

        out.println("ProblemIO");

    }
    catch( ServletException e ){

        out.println("ProblemServlet");
        out.println(e);

    }

}//end of sendtocreate


}//end of checkusername
```

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;

/**
 * This class invokes the CAPS compiler
 * @see java.servlet.http
 * @author J. A. McDonald III
 */
public class compile extends HttpServlet {

    String selectedProject = "";
    String selectedVersion = "";
    String userUserName = "";
    String mainClass = "";
    private String firstName;
    private String lastName;
    private String baseDirectory = "/c/caps1/Webapps/tomcat/webapps/CapsWeb/capsusers/";

    StringBuffer all;

    private PrintWriter  out;

    HttpSession session;

    HttpServletRequest request;
    HttpServletResponse response;


/**
* Processes the user request and responds via an HTML output
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
*/
public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException{

    res.setContentType("text/html");

    request = req;
    response = res;
    session=request.getSession();

    selectedProject = (String)session.getAttribute("selectedProject");
    selectedVersion = (String)session.getAttribute("selectedVersion");
    userUserName = (String)session.getAttribute("username");
    firstName = (String)session.getAttribute("firstname");
    lastName = (String)session.getAttribute("lastname");
    out = response.getWriter();

    topOfPage();

    compilePrototype();

    middleOfPage( all );

    bottomOfPage();

    out.close();

}//end of doGet
/**
 *Invokes the CAPS compiler
 */
public void compilePrototype(){

    String command = "/c/caps1/Webapps/tomcat/webapps/CapsWeb/bin/compile.script" +
```

55

```java
                " " + userUserName + " " + selectedProject + " " + selectedVersion;

        System.out.println(command);
        try{

            Runtime run = Runtime.getRuntime();
            run.exec(command);

        }
        catch(Exception ex ){

            out.println(ex);

        }


}//end translatePrototype()

/**
 *reads the log file
 */
public void readLogFile(){

    File file = new File( baseDirectory + userUserName + "/" + selectedProject +
                    "/" + selectedVersion + "/temp/" + selectedProject + ".log");
    String part;

    all = new StringBuffer("");

    if(file != null){

        try{

            FileReader fr = new FileReader(file);
            BufferedReader br= new BufferedReader(fr);
            while( (part=br.readLine()) != null){

                all.append(part);
                all.append('\n');
            }
            br.close();
            fr.close();

        }
        catch( IOException fnf ){

            out.println( fnf );

         }

        }
        else{

            out.println( "File not found.");

        }

}//end of readLogFile()

/**
 *Reads the info File
 */
public void readInfoFile(){

    File file = new File( baseDirectory + userUserName + "/" + selectedProject +
                        "/" + selectedVersion + "/temp/compile.info");
    String part;
```

56

```java
        all = new StringBuffer("");

        if(file != null){

            try{

                FileReader fr = new FileReader(file);
                BufferedReader br= new BufferedReader(fr);

                while( (part=br.readLine()) != null){

                    mainClass = part;
                }

                br.close();
                fr.close();

                }
                catch( IOException fnf ){

                    out.println( fnf );

                }

            }
            else{

                out.println( "File not found.");

            }


}//end of readLogFile()
/**
 *Outputs the top of the HTML Page
 */
private void topOfPage(){

    out.println("<html><head><title>Distributed Computer Aided Prototyping
System</title>");
    out.println("<style fprolloverstyle>A:hover {color: #FF0000; font-size: 14pt;
vertical-align: 10; font-weight: bold}</style>");
    out.println("</head><hr color="+"\"#0000FF\">");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>Distributed Computer Aided Prototyping
System</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>Web-CAPS Complilation</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>User:  "+firstName+" "+lastName+"</i></b></font></p>");

    out.println("<hr color="+"\"#0000FF\"><br>");

    out.println("<body background=\"/CapsWeb/bg2.gif\">");

}

/**
 *Outputs the middle of the HTMl Page
 */
private void middleOfPage( StringBuffer all){

    out.println("<table border="+"\"0\"  align="+"\"center\" cellpadding="+"\"4\" >");

    out.println("<tr>");

    out.println("<td  align="+"\"center\" ><font size="+"\"4\" color="+
    "\"#0000FF\"><a href="+"\"/CapsWeb/capsusers/"+userUserName+"/"+
    selectedProject+"/"+selectedVersion+
```

57

```
      "/temp/compile.err\" target=\"Resource Window\">View Compilation
Errors</a></font></td>");
      out.println("<td  align="+"\"center\" ><font size="+"\"4\" color="+
      "\"#0000FF\"><a href="+"\"/CapsWeb/capsusers/"+userUserName+"/"+selectedProject+
      "/"+selectedVersion+
      "/temp/compile.trace\" target=\"Resource Window\">View Compilation
Log</a></font></td>");
      out.println("<td  align="+"\"center\" ><font size="+"\"4\" color="+
      "\"#0000FF\"><a href="+"\"/CapsWeb/servlet/manager\">Return to
Manager</a></font></td>");

      out.println("</tr>");
      out.println("</table>");


}
/**
 *Outputs the bottom of the HTML Page
 */
private void bottomOfPage(){

      out.println("<br><br><br>"+"<hr color="+"\"#0000FF\">");
      out.println("<center><a href="+"\"/CapsWeb/servlet/logout\">Logout</a></center>");
      out.println("</html>");


}


}
```

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;


/**
 * This class checks the usernames in the database for uniquness
 *
 * @see javax.servlet.http
 * @author J. A. McDonald
 */
public class createaccount extends HttpServlet {

    private String userFirstName;
    private String userLastName;
    private String userUserName;
    private String userPassWord;

    //private String baseDirectory = "\\Program
Files\\Allaire\\JRun\\servers\\default\\CapsWeb\\capsUsers";
    private String baseDirectory = "/c/caps1/Webapps/tomcat/webapps/CapsWeb/capsusers";

    private Timestamp dateoflogin;

    private String driverName = "org.gjt.mm.mysql.Driver";
    private String urlName = "jdbc:mysql://131.120.9.87:3306/capsdata";


    private Driver        driver;

    private Connection    connection;

    private PrintWriter   out;

    HttpSession session;
/**
 * Processes the user request and responds via an HTML output
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
 */
public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException{

    res.setContentType("text/html");

    session = req.getSession();

    userFirstName = (String)session.getAttribute( "userfirstname" );
    userLastName = (String)session.getAttribute( "userlastname" );
    userPassWord = (String)session.getAttribute( "userpassword" );
    userUserName = (String)session.getAttribute( "userusername" );

    // Gets the PrintWriter object for sending HTML commands
    out = res.getWriter();

    //Generates the body
    establishConnection( );

    addUser();

    createDirectory();

}//end of doGet


/**
 * Establishes a connection to the database
```

```
*/
private void establishConnection( ){

    Properties props = new Properties();

    props.put("user",     "mcdonald");
    props.put("password", "mcdonald");

    try {

        driver   = (Driver) Class.forName( driverName ).newInstance();
        onnection = driver.connect( urlName, props );

    }
    catch( Exception e ){

        out.println("<p> Error: Cannot establish a connection to the database");

    }

}//end of establishConnection

/**
 * Adds user to the database
 *
*/
private void addUser( ){

    Statement sqlStmt = null;

    dateoflogin = new Timestamp(System.currentTimeMillis());

    String inputtime = dateoflogin.toString();

    try {

        String query = "INSERT into user values ('"
+userUserName+"','"+userPassWord+"','"+
                      userFirstName+"','"+ userLastName+"','"+ inputtime+"')";

        sqlStmt = connection.createStatement( );

        sqlStmt.executeUpdate( query );

        connection.close();

        sqlStmt.close();

    }
    catch (Exception e){

        out.println(e);

    }

}//end of addUser

/**
 * Creates the directory of the user
 *
*/
private void createDirectory( ){

    String userDirectory = "";

    userDirectory = baseDirectory + File.separator + userUserName;

    try{
```

60

```
         File td = new File(userDirectory);

         td.mkdirs();

    }
    catch (Exception e){

         out.println(e);

    }

}//end of createDirectory


}//end of createaccount
```

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;

/**
 * This class creates an HTML form listing projects of the user and allows them
 * to add a designated project name
 * @see java.servlet.http
 * @author J. A. McDonald III
 */
public class createProject extends HttpServlet {

    private String userUserName;
    private String firstName;
    private String lastName;


    private Timestamp dateoflogin;

    private Driver       driver;

    private Connection   connection;

    private PrintWriter  out;

    Vector projectNames;

    Vector projectVersions;

    private RequestDispatcher rd;

    HttpSession session;

    HttpServletRequest request;
    HttpServletResponse response;

    boolean dbStatus = false;
/**
* Processes the user request and responds via an HTML output
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
*/
public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException{

    res.setContentType("text/html");

    request = req;
    response = res;
    out = res.getWriter();
    session = req.getSession();
    firstName = (String)session.getAttribute("firstname");
    lastName = (String)session.getAttribute("lastname");
    projectNames = (Vector)session.getAttribute("projectNames");
    projectVersions = (Vector)session.getAttribute("projectVersions");

    topOfPage();

    listProjects( projectNames, projectVersions );

    projectForm();

    bottomOfPage();

    out.close();


}
```

```java
/**
 *Outputs the project names and versions in HTML Format
 *@param project Vector
 *@param version Vector
 */
private void listProjects( Vector project, Vector version ){

    int j = 0;

    String tempName = "";
    String tempVer  = "";

    Enumeration e = project.elements();
    Enumeration f = version.elements();

    out.println("<table border="+"\"0\"  align="+"\"center\" cellspacing="+
    "\"8\"  cellpadding="+"\"0\" >");

    out.println("<tr>");

    out.println("<td  align="+"\"left\" ><font size="+"\"3\" color="+
    "\"#0000FF\"><u>Project</font></u></td>");

    out.println("<td  align="+"\"left\" ><font size="+"\"3\" color="+
    "\"#0000FF\"><u>Version</font></u></td>");

    out.println("</tr>");

    while( e.hasMoreElements() && f.hasMoreElements() ){

        tempName = (String)e.nextElement();
        tempVer  = (String)f.nextElement();

        out.println("<tr>");

        out.println("<td  align="+"\"left\" ><font size="+"\"3\" color="+"\"#0000FF\">"+
        tempName+"</font></td>");

        out.println("<td  align="+"\"left\" ><font size="+"\"3\" color="+"\"#0000FF\">"+
        tempVer +"</font></td>");

        out.println("</tr>");

        j++;


    }

    out.println("</table>");

    if( j == 0 ){

        out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
        "\"#0000FF\"><b>Currently you have no Projects in the System.</b></font></p>");

    }


    out.println("<br><hr color="+"\"#0000FF\"><br>");


}

/**
 *Creates a form for user to input project name and version
 */
private void projectForm(){
```

63

```java
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b>Enter a Project Name and Version</b></font></p>");

    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><form  onSubmit="+"\"return validateEntry(this)\" method ="+
    "\"get\" action ="+
    "\"/CapsWeb/servlet/processNewProject\"><font color=blue> New Project Name:<input
type=text size="+
    "\"25\" name ="+"\"projName\">");
    out.println("<form onSubmit="+"\"return validateEntry(this)\"  method ="+
    "\"get\" action ="+
    "\"/CapsWeb/servlet/processNewProject\"><font color=blue> New Project Version:<input
type=text size="+
    "\"4\" name ="+"\"projVer\"></font></p>");
    out.println("<center><input type="+"\"submit\" value="+
    "\"Create\">  <input type=reset></center>");

}
/**
 *Outputs the top of the HTML Page
 */
private void topOfPage(){

    out.println("<html><head><title>Distributed Computer Aided Prototyping
System</title>");
    setJavaScript();
    out.println("<style fprolloverstyle>A:hover {color: #FF0000; font-size: 14pt;
vertical-align: 10; font-weight: bold}</style>");
    out.println("</head><hr color="+"\"#0000FF\">");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>Distributed Computer Aided Prototyping
System</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>Project Manager</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>User:  "+firstName+" "+lastName+"</i></b></font></p>");
    out.println("<hr color="+"\"#0000FF\"><br>");
    out.println("<body background=\"/CapsWeb/bg2.gif\">");

}

/**
 *Outputs the bottom of the HTML Page
 */
private void bottomOfPage(){

    out.println("<br><br><br>"+"<hr color="+"\"#0000FF\">");
    out.println("<center><a href="+"\"/CapsWeb/servlet/logout\">Logout</a></center>");
    out.println("</html>");


}
/**
 *Sets Javascript for the output page
 */
private void setJavaScript(){

    out.println("<SCRIPT LANGUAGE="+"\"JavaScript\">");

    out.println("<!-- hide this script from old browsers");

    out.println("function validateEntry( form ) {");

    out.println("if (form.projName.value ==\""+"\") {");

    out.println("alert("+"\"You must enter a Project Name\");");

    out.println("form.projName.focus();");

    out.println("return false;");
```
64

```
    out.println("}");

    out.println("else if (form.projVer.value ==\""+"\") {");
    out.println("alert("+"\"You must enter a Project Version\");");

    out.println("form.projVer.focus();");
    out.println("return false;");
    out.println("}");
    out.println("else {");
    out.println("return true;");
    out.println("}");
    out.println("}");
    out.println("// end hiding the script from old browsers -->
");
    out.println("</SCRIPT>");


}//end of setJavaScript




}
```

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;


/**
 * This class creates an HTML form listing projects of the user and allows them
 * to add a designated project version
 * @see java.servlet.http
 * @author J. A. McDonald III
 */
public class createProjectVersion extends HttpServlet {

    private String userUserName;
    private String firstName;
    private String lastName;


    private Timestamp dateoflogin;

    private Driver        driver;

    private Connection    connection;

    private PrintWriter   out;

    Vector projectNames;

    Vector projectVersions;

    private RequestDispatcher rd;

    HttpSession session;

    HttpServletRequest request;
    HttpServletResponse response;

    boolean dbStatus = false;
/**
* Processes the user request and responds via an HTML output
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
*/
public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException{

    res.setContentType("text/html");

    request = req;
    response = res;

    out = res.getWriter();
    session = req.getSession();
    firstName = (String)session.getAttribute("firstname");
    lastName = (String)session.getAttribute("lastname");
    projectNames = (Vector)session.getAttribute("projectNames");
    projectVersions = (Vector)session.getAttribute("projectVersions");

    topOfPage();

    listProjects( projectNames, projectVersions );

    versionForm();

    bottomOfPage();

    out.close();
}
```

```java
/**
 *Outputs the project names and version in HTML format
 *@param project Vector
 *@param version Vector
 */
private void listProjects( Vector project, Vector version ){

    int j = 0;

    String tempName = "";
    String tempVer  = "";

    Enumeration e = project.elements();
    Enumeration f = version.elements();

    out.println("<table border="+"\"0\"  align="+"\"center\" cellspacing="+
    "\"8\"  cellpadding="+"\"0\" >");

    out.println("<tr>");

    out.println("<td  align="+"\"left\" ><font size="+"\"3\" color="+
    "\"#0000FF\"><u>Project</font></u></td>");

    out.println("<td  align="+"\"left\" ><font size="+"\"3\" color="+
    "\"#0000FF\"><u>Version</font></u></td>");

    out.println("</tr>");

    while( e.hasMoreElements() && f.hasMoreElements() ){

        tempName = (String)e.nextElement();
        tempVer  = (String)f.nextElement();
        out.println("<tr>");

        out.println("<td  align="+"\"left\" ><font size="+"\"3\" color="+
        "\"#0000FF\">"+tempName+"</font></td>");

        out.println("<td  align="+"\"left\" ><font size="+"\"3\" color="+
        "\"#0000FF\">"+tempVer +"</font></td>");

        out.println("</tr>");

        j++;


    }

    out.println("</table>");

    if( j == 0 ){

        out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
        "\"#0000FF\"><b>Currently you have no Projects in the System.</b></font></p>");

    }

    out.println("<br><hr color="+"\"#0000FF\"><br>");

}

/**
 *Creates HTML Form to allow user to input project name and versio
 */
private void versionForm(){

    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b>Enter a Project Name and the New Version</b></font></p>");

    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><form onSubmit="+"\"return validateEntry(this)\"  method ="+
```

67

```
        "\"get\" action ="+"\"/CapsWeb/servlet/processNewVersion\"><font color=blue> Project
Name: &nbsp<input type=text size="+"\"25\" name ="+"\"projName\">");
    out.println("<form onSubmit="+"\"return validateEntry(this)\"  method ="+
    "\"get\" action ="+
    "\"/CapsWeb/servlet/processNewVersion\"><font color=blue> New Project
Version: &nbsp<input type=text size="+
    "\"4\" name ="+"\"projVer\"></font></p>");
    out.println("<center><input type="+"\"submit\" value="+
    "\"Create\">  <input type=reset></center>");

}
/**
 *Outputs top of HTML page
 */
private void topOfPage(){

    out.println("<html><head><title>Distributed Computer Aided Prototyping
System</title>");
    setJavaScript();
    out.println("<style fprolloverstyle>A:hover {color: #FF0000; font-size: 14pt;
vertical-align: 10; font-weight: bold}</style>");
    out.println("</head><hr color="+"\"#0000FF\">");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>Distributed Computer Aided Prototyping
System</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>Project Manager</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>User:  "+firstName+" "+lastName+"</i></b></font></p>");

    out.println("<hr color="+"\"#0000FF\"><br>");

  out.println("<body background=\"/CapsWeb/bg2.gif\">");

}

/**
 *Outputs bottom of HTML Page
 */
private void bottomOfPage(){

    out.println("<br><br><br>"+"<hr color="+"\"#0000FF\">");
    out.println("<center><a href="+"\"/CapsWeb/servlet/logout\">Logout</a></center>");
    out.println("</html>");


}

/**
 *Sets the Javascript for the created page
 */
private void setJavaScript(){

    out.println("<SCRIPT LANGUAGE="+"\"JavaScript\">");

    out.println("<!-- hide this script from old browsers");

    out.println("function validateEntry( form ) {");

    out.println("if (form.projName.value ==\""+"\") {");

    out.println("alert("+"\"You must enter a Project Name\");");

    out.println("form.projName.focus();");
    out.println("return false;");

    out.println("}");
    out.println("else if (form.projVer.value ==\""+"\") {");

    out.println("alert("+"\"You must enter a Project Version\");");
```

```
    out.println("form.projVer.focus();");

    out.println("return false;");

    out.println("}");

    out.println("else {");

    out.println("return true;");

    out.println("}");

    out.println("}");

    out.println("// end hiding the script from old browsers -->
");

    out.println("</SCRIPT>");

}//end of setJavaScript



}
```

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;

/**
 * This class dynamically creates a .jnlp file in order to execute the prototype
 * developed by the user
 * @see java.servlet.http
 * @author J. A. McDonald III
 */
public class executejnlp extends HttpServlet {

    String selectedProject = "";
    String selectedVersion = "";
    String userUserName = "";
    String mainClass = "";
    private String baseDirectory = "/c/caps1/Webapps/tomcat/webapps/CapsWeb/capsusers/";

    private PrintWriter  out;

    HttpSession session;

    HttpServletRequest request;
    HttpServletResponse response;


/**
   * Responds to a request and responds
 *@see javax.servlet.http
   * @param HttpServletRequest req
   *@param HttpServletResponse res
   */
public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException{

    res.setContentType("application/x-java-jnlp-file");

    request = req;
    response = res;
    session=request.getSession();

    selectedProject = (String)session.getAttribute("selectedProject");
    selectedVersion = (String)session.getAttribute("selectedVersion");
    userUserName = (String)session.getAttribute("username");
    readInfoFile();

    out = response.getWriter();

    out.println("<?xml version="+"\"1.0\" encoding="+"\"utf-8\"?>");
    out.println("<!-- JNLP File for Prototype Execution Application -->");
    out.println("<jnlp");
    out.println("       spec="+"\"1.0+\"");
        //out.println("        codebase="+"\"http://131.120.9.87:8080/CapsWeb\">");
    out.println("      codebase="+"\"http://seaotter.cs.nps.navy.mil:8080/CapsWeb\">");

    out.println("      <information>");
    out.println("              <title>Distributed CAPS Application</title>");
    out.println("              <vendor>Software Engieering Group, Naval Postgraduate
School</vendor>");
    out.println("              <homepage href="+"\"WebCaps.htm\"/>");
    out.println("              <description>Prototype:  "+selectedProject+
    "  Version:  "+selectedVersion+" </description>");
    out.println("              <description kind="+"\"short\">The
Prototype.</description>");
    out.println("                  <icon href="+"\"/CapsWeb/globi.gif\"/>");
    out.println("              <offline-allowed/>");
    out.println("      </information>");
    out.println("      <security>");
```

```java
    out.println("               <all-permissions/>");
    out.println("       </security>");
    out.println("       <resources>");
    out.println("               <j2se version="+"\"1.2.2\"/>");
    out.println("               <jar href="+"\"/CapsWeb/capsusers/"+userUserName+
    "/"+selectedProject+"/"+selectedVersion+"/"+"bin/"+mainClass+".jar\" download="+
    "\"eager\"/>");
    out.println("       </resources>");
    out.println("<application-desc main-class=\""+mainClass+"\" >");
    out.println("<argument>"+mainClass+"</argument>");

    out.println("       <application-desc/>");

    out.println("</jnlp>");

    out.close();

}

/**
 *read the compile.info file to get the main class of the prototype
 */
public void readInfoFile(){

    File file = new File( baseDirectory + userUserName + "/" + selectedProject +
                          "/" + selectedVersion + "/temp/compile.info");
    String part;

        //all = new StringBuffer("");

    if(file != null){

        try{

            FileReader fr = new FileReader(file);
            BufferedReader br= new BufferedReader(fr);
            while( (part=br.readLine()) != null){

                mainClass = part;
            }
            br.close();
            fr.close();

        }
        catch( IOException fnf ){

            out.println( fnf );

        }

    }

    else{

        out.println( "File not found.");

    }

}//end of readLogFile()


}
```

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;

/**
 * This class checks the usernames in the database for uniquness
 *
 * @see javax.servlet.http
 * @author J. A. McDonald
 */
public class logonadmin extends HttpServlet{

    private String adminUserName;
    private String adminPassWord;
    private String password;
    private String firstname;
    private String lastname;

    private Timestamp dateoflogin;

    private String driverName = "org.gjt.mm.mysql.Driver";
    private String urlName = "jdbc:mysql://131.120.9.87:3306/capsdata";

    private Driver        driver;

    private Connection    connection;

    private PrintWriter  out;

    HttpSession session;

    HttpServletRequest request;
    HttpServletResponse response;

    boolean dbStatus = false;
/**
* Processes the user request and responds via an HTML output
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
*/
public void doPost(HttpServletRequest req, HttpServletResponse res)throws IOException{

    res.setContentType("text/html");

    request = req;
    response = res;

    adminUserName = request.getParameter("username");
    adminPassWord = request.getParameter("password");

    out = res.getWriter();

    admincheckIn( );

}//end of doPost

/**
* Processes the user request and responds via an HTML output and calls doPost
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
*/
public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException{

    doPost(req,res);
```

72

```
}//end of doGet

/**
* Check to see if the given username/password is a valid one in the database
*/
private void admincheckIn( ){

    dbStatus = establishConnection( );//attempt to connect to database

    if( dbStatus ){//if connection if good then check on username/password

        replyClient( request, response );

    }
    else{//if connection to database is no good

        topOfPage();

        out.println("<center>Attempted to check in  " + adminUserName +
                    "  at " + new java.util.Date() + "</h3><p>" );

        out.println("Unable to connect to the Database at this time.</p></center>");

    }

}//end of admincheckIn

/**
 * Establishes a connection to the database
 * @return a <code>boolean</code> true if successful, otherwise false
*/
private boolean establishConnection( ){

    Properties props = new Properties();

    props.put("user",     "mcdonald");
    props.put("password", "mcdonald");

    try{
        driver   = (Driver) Class.forName( driverName ).newInstance();
        connection = driver.connect( urlName, props );
        return true;

    }
    catch( Exception e ){

        out.println(e);

          return false;

    }
}//end of establishConnection


/**
* Checks database for a valid username and password combo
*
*/
private void replyClient( HttpServletRequest req, HttpServletResponse res){

    Statement sqlStmt = null;
    ResultSet rs      = null;

    try{

      String query = "select fname,lname,password from administrator where username =" +
                    "'" + adminUserName + "'" ;

      sqlStmt = connection.createStatement( );
```

```java
        rs       = sqlStmt.executeQuery( query );

        if ( rs.next() ) {//if there is a match then return info for use

            firstname = rs.getString("fname");
            lastname = rs.getString("lname");
            password = rs.getString("password");

            if( password.equals(adminPassWord) ){

                connection.close();
                sqlStmt.close();
                success( req, res );

            }
            else{

                connection.close();
                sqlStmt.close();
                unsuccessful( req, res );

            }


        }
        else {//if the combo is not in the database then tell user

            topOfPage();

            out.println("<p align="+"\"center\"><font size="+
                "\"5\" color="+"\"#FF0000\"><b><i>Attempted to login  "+
                adminUserName + " at " + new java.util.Date() +
                "</i></b></font></p>");
            out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
                "\"#FF0000\"><b><i>Sorry, there doesn't seem to be an acount with that
username.  Contact the Caps Group. </i></b></font></p>");
            out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
                "\"#FF0000\"><b><i>Or attempt to login again. </i></b></p>");
            out.println("<center><a href="+
                "\"/CapsAdmin/adminlogin.html\">Login</a></font></center>");

            bottomOfPage();

        }
        //close connection to the database
        connection.close();
        sqlStmt.close();

    }

    catch (Exception e){

    }

}//end of replyClient

/**
* Outputs an HTML if login successful
 * @throws IOException
*
*/
private void success(HttpServletRequest req, HttpServletResponse res)throws IOException{

    session = req.getSession(true);//set the session and set attributes

    topOfPage();
```

```java
    out.println("<p align="+"\"center\"><font size="+"\"4\" color="+
            "\"#0000FF\"><b><i>Administrator:  "+ firstname +
            " " + lastname + "</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"4\" color="+
            "\"#0000FF\"><b><i>Logged-in on:  " + new java.util.Date() +
            "</i></b></font></p>");

    middleOfPage();

    bottomOfPage();

    setDate();
}//end of successful

/**
* Outputs and HTML page if login is unsuccessful
 *@throws IOExecption
*
*/
private void unsuccessful(HttpServletRequest req, HttpServletResponse res)throws
IOException{

    topOfPage();

    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
            "\"#FF0000\"><b><i>Your login attempt was unsuccessful.</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
            "\"#FF0000\"><b><i>Either your username, password or both are
incorrect.</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
            "\"#FF0000\"><b><i>Please Try Again.</i></b></font></p>");
    out.println("<center><a href="+"\"/CapsAdmin/adminlogin.html\">Login</a></center>");

    bottomOfPage();

}//end of unsuccessful


/**
* Outputs the top of the HTML Page
*
*/
private void topOfPage(){

        out.println("<html><head><title>Distributed Computer Aided Prototyping
System</title>");
        out.println("<style fprolloverstyle>A:hover {color: #FF0000; font-size: 14pt;
vertical-align: 10; font-weight: bold}</style>");
        out.println("</head><hr color="+"\"#0000FF\">");
        out.println("<p align="+"\"center\"><font size="+"\"5\"
color="+"\"#0000FF\"><b><i>Distributed Computer Aided Prototyping
System</i></b></font></p>");
        out.println("<p align="+"\"center\"><font size="+"\"5\"
color="+"\"#0000FF\"><b><i>Administrator Tools</i></b></font></p>");

        out.println("<hr color="+"\"#0000FF\"><br>");

}//end of topOfPage

/**
* Outputs the middle of the HTML Page
*
*/
private void middleOfPage(){

        out.println("<body>");

        out.println("<p align="+"\"center\"><font size="+"\"4\" color="+
                "\"#0000FF\"><b><i>Date of your last login was "+getDate()+
```

75

```java
                "</i></b></font></p>");

        out.println("<table border="+"\"1\"  align="+"\"center\" cellpadding="+
            "\"4\" >");

        out.println("<tr>");

        out.println("<td  align="+"\"center\" ><font size="+"\"4\" color="+
            "\"#0000FF\"><a href="+
            "\"/CapsAdmin/createaccountattemp.html\">Create User
Account</a></font></td>");

        out.println("</tr>");

        out.println("</table>");

        out.println("</body>");


}//end of middleOfPage

/**
* Outputs the bottom of the HTML Page
*
*/
private void bottomOfPage(){

        out.println("<br><br><br>"+"<hr color="+"\"#0000FF\">");
        out.println("<center><a
href="+"\"/CapsAdmin/servlet/logoutAdmin\">Logout</a></center>");
        out.println("</html>");


}//end of bottomOfPage

/**
* Updates the date of last login in the database
*
*/
private void setDate(){

    dbStatus = establishConnection();

    dateoflogin = new Timestamp(System.currentTimeMillis());

    String input = dateoflogin.toString();


    if( dbStatus ){

        Statement sqlStmt = null;
        ResultSet rs      = null;

        try {

                String    query   = "update administrator set
administrator.datestamp='"+input+
                "' where administrator.username =" + "'" + adminUserName + "'";

                sqlStmt = connection.createStatement( );

                sqlStmt.executeUpdate( query );

                connection.close();
                sqlStmt.close();

        }

        catch(Exception e){
```

76

```java
                        out.println(e);

                }



        }
        else{

            out.println("unable to connect to database");

        }

}//end of setDate

/**
* @return a <code>String</code> specifying the last login date from the database
*
*/
private String getDate(){

    String timeoflastlogin = "";

    dbStatus = establishConnection();

    if( dbStatus ){

        Statement sqlStmt = null;
        ResultSet rs      = null;

        try {

                String  query = "select datestamp from administrator where username =" +
                    "'" + adminUserName + "'" ;

                sqlStmt = connection.createStatement( );

                rs      = sqlStmt.executeQuery( query );

                while( rs.next() ){

                        timeoflastlogin = rs.getString("datestamp");

                }

                connection.close();
                sqlStmt.close();

        }

        catch(Exception e){

                out.println(e);

        }

        return timeoflastlogin;


    }
    else{

        String problemwithdb = "Unable to connect to database";

        return problemwithdb;

    }
```

```
}//end of getDate

}//end of logonadmin
```

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;


/**
 * This class connects to database and checks to see if the username/password
 * matches,  if so they are logged in, if not they are denied.  A session is
 * begun at this time.
 * @see java.servlet.http
 * @author J. A. McDonald III
 */
public class logonuser extends HttpServlet {

    private String userUserName;
    private String userPassWord;
    private String password;
    private String firstName;
    private String lastName;
    private String accessAmt;

    private String driverName = "org.gjt.mm.mysql.Driver";
    private String urlName = "jdbc:mysql://131.120.9.87:3306/capsdata";


    private Timestamp dateoflogin;

    private Driver        driver;

    private Connection    connection;

    private PrintWriter   out;

    private RequestDispatcher rd;

    HttpSession session;

    HttpServletRequest request;
    HttpServletResponse response;

    boolean dbStatus = false;
/**
* Processes the user request and responds via an HTML output
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
*/
public void doPost(HttpServletRequest req, HttpServletResponse res)throws IOException{

    res.setContentType("text/html");

    request = req;
    response = res;
    userUserName = request.getParameter("username");
    userPassWord = request.getParameter("password");

    out = res.getWriter();

        //Check the database

    usercheckIn( );

    out.close();

}
/**
* Processes the user request and responds via an HTML output
 *@param HttpServletRequest req
```

```
 *@param HttpServletResponse res
 *@throws IOException
*/
public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException{

    doPost( req, res );

}


  /**
   * Check to see if the given username/password is a valid one in the database
   */
private void usercheckIn( ){

    dbStatus = establishConnection( );//attempt to connect to database

    if( dbStatus ){//if connection if good then check on username/password

        replyClient( request, response );

    }
    else{//if connection to database is no good

        topOfPage();

        out.println("<center>Attempted to check in  " + userUserName +
                    "  at " + new java.util.Date() + "</h3><p>" );

        out.println("Unable to connect to the Database at this time.</p></center>");

     }


}
  /**
   * Establishes a connection to the database
   *@return <code>boolean</code>
   */
private boolean establishConnection( ){

    Properties props = new Properties();

    props.put("user",     "mcdonald");
    props.put("password", "mcdonald");

    try {

        driver   = (Driver) Class.forName( driverName ).newInstance();
        connection = driver.connect( urlName, props );
        return true;

    }
    catch ( Exception e ) {

        return false;

    }
}
  /**
   * Checks database for a valid username and password combo
   * @param HttpServletRequest req
   *@param HttpServletResponse res
   */
private void replyClient( HttpServletRequest req, HttpServletResponse res){

    Statement sqlStmt = null;
```

80

```java
    ResultSet rs      = null;

    try {

        String    query   = "select fname, lname, password from user where username =" +
        "'" + userUserName + "'" ;

        sqlStmt = connection.createStatement( );

        rs       = sqlStmt.executeQuery( query );


        if ( rs.next() ) {//if there is a match then return info for use

            firstName = rs.getString("fname");
            lastName  = rs.getString("lname");
            password  = rs.getString("password");


        if( password.equals(userPassWord) ){

                connection.close();
                sqlStmt.close();
                success( req, res );

        }

        else{

                connection.close();
                sqlStmt.close();
                unsuccessful( req, res );

        }


        }

        else {//if the combo is not in the database then tell user

            topOfPage();
            out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
            "\"#FF0000\"><b><i>Attempted to login  "+ userUserName + " at " +
            new java.util.Date() + "</i></b></font></p>");
            out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
            "\"#FF0000\"><b><i>Sorry, there doesn't seem to be an account with that
username.  Contact the Caps Group. </i></b></font></p>");
            out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
            "\"#FF0000\"><b><i>Or attempt to login again. </i></b></p>");
            out.println("<center><a
href="+"\"/CapsWeb/userlogin.html\">Login</a></font></center>");
            out.println("<br><br><br>"+"<hr color="+"\"#0000FF\">");
            out.println("</html>");


        }
            //close connection to the database
            connection.close();
            sqlStmt.close();

    }

    catch (Exception e) { }

  }

   /**
```

```java
    * Begins session and logs user in if authorized
    * @param HttpServletRequest req
    *@param HttpServletResponse res
    */

private void success(HttpServletRequest req, HttpServletResponse res)throws IOException{

    session = req.getSession(true);//set the session and set attributes
    session.setAttribute("username", userUserName);
    session.setAttribute("firstname", firstName);
    session.setAttribute("lastname", lastName);

    topOfPage();
    out.println("<p align="+"\"center\"><font size="+"\"4\" color="+
    "\"#0000FF\"><b><i>User:  "+ firstName + " " + lastName + "</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"4\" color="+
    "\"#0000FF\"><b><i>Logged-in on:  " + new java.util.Date() + "</i></b></font></p>");

    middleOfPage();

    bottomOfPage();

    setDate();
}

/**
    * If user not authorized it outputs an HTML Page saying so
    * @param HttpServletRequest req
    *@param HttpServletResponse res
    */
private void unsuccessful(HttpServletRequest req, HttpServletResponse res)throws
IOException{

    topOfPage();

    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#FF0000\"><b><i>Your login attempt was unsuccessful.</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#FF0000\"><b><i>Either your username, password or both are
incorrect.</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#FF0000\"><b><i>Please Try Again.</i></b></font></p>");
    out.println("<center><a href="+"\"/CapsWeb/userlogin.html\">Login</a></center>");

    bottomOfPage();
}
/**
 *Outputs top of HTML Page
 */
private void topOfPage(){

    out.println("<html><head><title>Distributed Computer Aided Prototyping
System</title>");
    out.println("<style fprolloverstyle>A:hover {color: #FF0000; font-size: 14pt;
vertical-align: 10; font-weight: bold}</style>");
    out.println("</head><hr color="+"\"#0000FF\">");
    out.println("<p align="+"\"center\"><font size="+
    "\"5\" color="+"\"#0000FF\"><b><i>Distributed Computer Aided Prototyping
System</i></b></font></p>");
    out.println("<hr color="+"\"#0000FF\"><br>");
    out.println("<body background=\"/CapsWeb/bg2.gif\">");
}

/**
 *Outputs middle of the HTML Page
 */
private void middleOfPage(){

    out.println("<p align="+"\"center\"><font size="+"\"4\" color="+
```

```java
    "\"#0000FF\"><b><i>Date of your last login was "+getDate()+"</i></b></font></p>");
    out.println("<table border="+"\"0\"  align="+"\"center\" cellpadding="+"\"4\" >");
    out.println("<tr>");

    out.println("<td  align="+"\"center\" ><font size="+"\"4\" color="+
    "\"#0000FF\"><a href="+"\"/CapsWeb/servlet/projectManager\">Project
Manager</a></font></td>");
    out.println("</tr>");
    out.println("</table>");


}

/**
 *Outputs the bottom of the HTML Page
 */
private void bottomOfPage(){

    out.println("<br><br><br>"+"<hr color="+"\"#0000FF\">");
    out.println("<center><a href="+"\"/CapsWeb/servlet/logout\">Logout</a></center>");
    out.println("</html>");


}
/**
 *sets the logon time of the user
 */
private void setDate(){

    dbStatus = establishConnection();

    dateoflogin = new Timestamp(System.currentTimeMillis());

    String input = dateoflogin.toString();


    if( dbStatus ){

        Statement sqlStmt = null;
        ResultSet rs      = null;

        try {

                String   query   = "update user set user.datestamp='"+input+
                "' where user.username =" + "'" + userUserName + "'";

                sqlStmt = connection.createStatement( );

                sqlStmt.executeUpdate( query );

                connection.close();
                sqlStmt.close();

        }

        catch(Exception e){

                out.println(e);

        }


    }


    else{
```

83

```java
        out.println("unable to connect to database");


        }

}

/**
 *retrieves the last login time of the user from the database
 */
private String getDate(){

    String timeoflastlogin = "";

    dbStatus = establishConnection();

    if( dbStatus ){

        Statement sqlStmt = null;
        ResultSet rs      = null;

        try {

                String    query  = "select datestamp from user where username =" +
                "'" + userUserName + "'" ;

                sqlStmt = connection.createStatement( );

                rs      = sqlStmt.executeQuery( query );

                while( rs.next() ){

                        timeoflastlogin = rs.getString("datestamp");

                }

                connection.close();
                sqlStmt.close();

        }

        catch(Exception e){

                out.println(e);

        }

        return timeoflastlogin;

    }

    else{

        String problemwithdb = "Unable to connect to database";

        return problemwithdb;

    }

}

}
```

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;

/**
 * This class destroys the session of the user and logs them out
 * @see java.servlet.http
 * @author J. A. McDonald III
 */
public class logout extends HttpServlet {

    String selectedProject = null;


    private Driver        driver;

    private Connection    connection;

    private PrintWriter  out;



    HttpSession session;

    HttpServletRequest request;
    HttpServletResponse response;


/**
* Processes the user request and responds via an HTML output
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
*/
public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException{

    res.setContentType("text/html");

    request = req;
    response = res;

    selectedProject = request.getParameter("selectedProject");
        // Gets the PrintWriter object for sending HTML commands

    out = response.getWriter();
    session = req.getSession();
    session.invalidate();

    topOfPage();
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b>You are now logged off.</b></p>");
    out.println("<center><a href="+
    "\"/CapsWeb/WebCAPS.htm\">Return to WebCaps HomePage.</a>");
    out.println("</font></center>");
    bottomOfPage();
    out.close();


}

/**
 *Outputs the top of the HTML Page
 */
private void topOfPage(){

    out.println("<html><head><title>Distributed Computer Aided Prototyping
System</title>");
```

85

```java
    out.println("<style fprolloverstyle>A:hover {color: #FF0000; font-size: 14pt;
vertical-align: 10; font-weight: bold}</style>");
    out.println("</head><hr color="+"\"#0000FF\">");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>Distributed Computer Aided Prototyping
System</i></b></font></p>");
    out.println("<hr color="+"\"#0000FF\"><br>");
    out.println("<body background=\"/CapsWeb/bg2.gif\">");

}

/**
 *Outputs the bottom of the HTML Page
 */
private void bottomOfPage(){

    out.println("<br><br><br>"+"<hr color="+"\"#0000FF\">");
    out.println("</html>");


}


}
```

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;

/**
 * This class destroys the session of the administrator logged in
 *
 * @see javax.servlet.http
 * @author J. A. McDonald
 */
public class logoutAdmin extends HttpServlet{

    String selectedProject = null;

    private String driverName = "org.gjt.mm.mysql.Driver";

    private String urlName = "jdbc:mysql://131.120.9.87:3306/capsdata";

    private Driver       driver;

    private Connection   connection;

    private PrintWriter  out;

    HttpSession session;

    HttpServletRequest request;

    HttpServletResponse response;


/**
* Processes the user request and responds via an HTML output
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
*/
public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException{

    res.setContentType("text/html");

    request = req;

    response = res;

    out = response.getWriter();

    session = request.getSession();

    session.invalidate();

    topOfPage();

    out.println("<p align="+"\"center\"><font size="+"\"5\" color="
                +"\"#0000FF\"><b>You are now logged off.</b></p>");

    out.println("<center><a href="+
        "\"/CapsAdmin/admin.html\">Return to WebCaps Admin Tools Page.</a>");

    out.println("</font></center>");

    bottomOfPage();

    out.close();

}//end of doGet

/**
```

```
 * Outputs the top portion of the HTML Page
 */
private void topOfPage(){

    out.println("<html><head><title>Distributed Computer Aided Prototyping System Admin
Tools</title>");
    out.println("<style fprolloverstyle>A:hover {color: #FF0000; font-size: 14pt;
vertical-align: 10; font-weight: bold}</style>");
    out.println("</head><hr color="+"\"#0000FF\">");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="
        +"\"#0000FF\"><b><i>Distributed Computer Aided Prototyping
System</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
            "\"#0000FF\"><b><i>Admin Tools</i></b></font></p>");
    out.println("<hr color="+"\"#0000FF\"><br>");

}//end of topOfPage

/**
 * Outputs the bottom portion of the HTML page
 *
 */
private void bottomOfPage(){

    out.println("<br><br><br>"+"<hr color="+"\"#0000FF\">");
    out.println("</html>");

}//end of bottomOfPage


}//end of logoutAdmin
```

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;

/**
 * This class creates an HTML output allowing users to choose b/n options
 * allowing them to develop the prototypes such as launching the editor
 * translating, scheduler, compiling, and execution as well as changing the project
 * to be worked on.  This is the second of two managers
 * @see projectManager.class
 * @see java.servlet.http
 * @author J. A. McDonald III
 */
public class manager extends HttpServlet {


    String selectedVersion = null;
    String selectedProject = null;
    String username = null;


    private Driver        driver;

    private Connection    connection;

    private PrintWriter   out;



    HttpSession session;

    HttpServletRequest request;
    HttpServletResponse response;

/**
* Processes the user request and responds via an HTML output
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
*/
public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException{

    res.setContentType("text/html");

    request = req;
    response = res;
    out = response.getWriter();

    session = request.getSession();
    selectedVersion = (String)session.getAttribute( "selectedVersion");
    selectedProject = (String)session.getAttribute("selectedProject");
    username = (String)session.getAttribute("username");

    topOfPage();

    out.println("<p align="+"\"center\"><font size="+"\"3\" color="+
    "\"#0000FF\"><b>User--"+username+"  Project--"+selectedProject+
    "  Version--"+selectedVersion+".</b></font></p>");
    out.println("<center><font size="+"\"4\" color="+"\"#0000FF\"><b><a href="+
    "\"/CapsWeb/servlet/capsjnlp\">Launch Editor</a>  ");
    out.println("<a href="+"\"/CapsWeb/servlet/upLoad\">Upload Project
Files</a>  ");
    out.println("<a href="+"\"/CapsWeb/servlet/translate\">Translate</a>  ");

    out.println("<a href="+"\"/CapsWeb/servlet/scheduler\">Schedule</a>  ");
    out.println("<a href="+"\"/CapsWeb/servlet/compile\">Compile</a>  ");
    out.println("<a href="+"\"/CapsWeb/servlet/executejnlp\">Execute</a>  ");
    out.println("<a href="+
```

89

```
    "\"/CapsWeb/servlet/setProject\">Reset Project</a></b></font></center>");
    bottomOfPage();
    out.close();


}

/**
 *Outputs the top of the HTML Page
 */
private void topOfPage(){

    out.println("<html><head><title>Distributed Computer Aided Prototyping
System</title>");
    out.println("<style fprolloverstyle>A:hover {color: #FF0000; font-size: 14pt;
vertical-align: 10; font-weight: bold}</style>");
    out.println("</head><hr color="+"\"#0000FF\">");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>Distributed Computer Aided Prototyping
System</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>Project Manager</i></b></font></p>");

    out.println("<hr color="+"\"#0000FF\"><br>");

    out.println("<body background=\"/CapsWeb/bg2.gif\">");

}

/**
 *Outputs the bottom of the HTML Page
 */
private void bottomOfPage(){

    out.println("<br><br><br>"+"<hr color="+"\"#0000FF\">");
    out.println("<center><a href="+"\"/CapsWeb/servlet/logout\">Logout</a></center>");
    out.println("</html>");


}


}
```

```java
import java.io.*;
import java.util.*;
import javax.servlet.*;
/**
 * This class uploads files identified by the user
 *
 *
 */
public class MultipartRequest {

private static final int DEFAULT_MAX_POST_SIZE = 1024 * 1024;

private ServletRequest req;
private File dir;
private int maxSize;

private Hashtable parameters = new Hashtable();
private Hashtable files = new Hashtable();

public MultipartRequest(ServletRequest request,
                        String saveDirectory) throws IOException {

  this(request,saveDirectory,DEFAULT_MAX_POST_SIZE);

}


public MultipartRequest(ServletRequest request,
                        String saveDirectory,
                        int maxPostSize) throws IOException {

//sanity check values

        if(request == null)
          throw new IllegalArgumentException(" request cant be null");

        if(saveDirectory == null)
          throw new IllegalArgumentException("saveDirectory cant be null");

        if(maxPostSize <= 0)
          throw new IllegalArgumentException("maxPostsize must be positive");


//save the request,dir and maxsize

req=request;

dir=new File(saveDirectory);

maxSize=maxPostSize;

//check saveDirectory is truely a directory

        if(!dir.isDirectory())
         throw new IllegalArgumentException("not a directory: " + saveDirectory);

//check saveDirectory is writable

        if(!dir.canWrite())
          throw new IllegalArgumentException("not writable:" + saveDirectory );

//now parse the request saving data to "parameters" and "files";
//write the file contents to saveDirectory

        readRequest();

}
```

```java
public Enumeration getParameterNames() {

        return parameters.keys();

}


public Enumeration getFileNames() {

        return files.keys();

}


public String getParameter(String name) {

        try{

                String param = (String)parameters.get(name);
                if(param.equals(""))return null;
                return param;

        }
        catch(Exception e) {

                return null;

        }

}


public String getFilesystemName(String name) {

        try {

                UploadedFile file = (UploadedFile)files.get(name);
                return file.getFilesystemName();

        }
        catch(Exception e) {

                return null;

        }

}


public String getContentType(String name) {

        try {

                UploadedFile file = (UploadedFile)files.get(name);
                return file.getContentType();

        }
        catch(Exception e) {

                return null;

        }

}

public File getFile(String name) {
```

```java
        try {

                UploadedFile file=(UploadedFile)files.get(name);
                return file.getFile();

        }
        catch(Exception e) {

                return null;

        }

}


protected void readRequest() throws IOException {

        //check the content type to make sure it's "multipart/form-data"
        String type=req.getContentType();

        if(type == null || !type.toLowerCase().startsWith("multipart/form-data")) {

                throw new IOException("posted content type isn't multipart/form-data");

        }

        //check the content length to prevent denial of service attacks
        int length = req.getContentLength();

        if(length > maxSize) {

                throw new IOException("posted content length of " + length +
                                "exceeds limit of " + maxSize);

        }

        //get the boundary string ;it's included in the content type.
        //should look something like "-----------------------12012133613061"

        String boundary=extractBoundary(type);

        if(boundary == null) {

                throw new IOException("seperation boundary was not specified");

        }


        //construct the special input stream we'll read from
        MultipartInputStreamHandler in = new
MultipartInputStreamHandler(req.getInputStream(),boundary,length);

        //read the first line ,should be the first boundary
        String line =in.readLine();

        if(line == null) {

                throw new IOException("corrupt form data: premature ending");

        }

        //verify that the line is the boundary
        if(!line.startsWith(boundary)) {

                throw new IOException("corrupt form data: no leading boundary");

        }
```

```java
        //now that we are just beyond the first boundary, loop over each part
        boolean done=false;

        while(!done) {

                done=readNextPart(in,boundary);

        }


}


protected boolean readNextPart(MultipartInputStreamHandler in, String boundary)throws
IOException {

        //read the first line ,should look like this
        //content-disposition:form-data;name="field1";filename="file1.txt"

        String line=in.readLine();

        if(line == null) {

                //no parts left,we'r done
                return true;

        }

        //parse the content-disposition line

        String[] dispInfo=extractDispositionInfo(line);
        String disposition=dispInfo[0];
        String name=dispInfo[1];
        String filename=dispInfo[2];

        //now onto the next line.this will either be empty
        //or contain a content-type and an empty line.

        line=in.readLine();
        if(line == null) {

                //no parts left,we/r done
                return true;

        }

        //get the content type.or null if none is specified
        String contentType=extractContentType(line);

        if(contentType != null) {

                //eaat the empty line
                line=in.readLine();

                if(line == null||line.length() > 0) {

                        //line should be empty
                        throw new IOException("malformed line after content type: " +
line);
                }
        }
        else {
                //assume a default content type
                contentType = "application/octet-stream";

        }
```

```java
        //now,finally we read the content (end after reading the boundary)

        if(filename == null) {

                //this is a parameter
                String value=readParameter(in,boundary);
                parameters.put(name,value);

        }
        else {

                //this is a file
                readAndSaveFile(in,boundary,filename);

                if(filename.equals("unknown")) {

                        files.put(name,new UploadedFile(null,null,null));

                }
                else {

                        files.put(name,new
UploadedFile(dir.toString(),filename,contentType));

                }
        }

        return false;

}


protected String readParameter(MultipartInputStreamHandler in, String boundary) throws
IOException {

         StringBuffer sbuf = new StringBuffer();
         String line;

         while((line = in.readLine()) != null) {

                if(line.startsWith(boundary)) break;
                sbuf.append(line + "\r\n");
                //add the \r\n in case there are many lines

         }

         if(sbuf.length() == 0) {

                return null;

         }

         sbuf.setLength(sbuf.length() - 2);//cut off the last line's \r\n
         return sbuf.toString();   //no URL decoding is needed

}


protected void readAndSaveFile(MultipartInputStreamHandler in, String boundary, String
filename) throws IOException {

        File f=new File(dir + File.separator + filename);

        FileOutputStream fos=new FileOutputStream(f);
```

```
        BufferedOutputStream out=new BufferedOutputStream(fos,8 * 1024);

        byte[]  bbuf=new byte[8 * 1024];

        int result;

        String line;

        //ServletInputSteam.readLine() has the annoying habit of
        //adding a \r\n to the end of the last line
        //since we want a byte-for-byte transfer,we have to cut those chars

        boolean rnflag= false;

        while((result=in.readLine(bbuf,0,bbuf.length)) != -1 ) {

                //check for the boundary

                if(result > 2 && bbuf[0] == '-' && bbuf[1] == '-') {

                        line=new String(bbuf,0,result,"ISO-8859-1");

                        if(line.startsWith(boundary))break;

                }

                //are we suppose to write \r\n for the last iteration ?
                if(rnflag) {

                        out.write('\r');
                        out.write('\n');
                        rnflag=false;

                }

                //write the buffer ,postpone any editing \r\n\
                if(result >= 2 && bbuf[result - 2 ] == '\r' && bbuf[result - 1] == '\n' )
{

                        out.write(bbuf,0,result - 2);
                        rnflag=true;

                }
                else{

                        out.write(bbuf,0,result);

                }
         }

         out.flush();
         out.close();
         fos.close();

}

private String extractBoundary(String line){

        int index=line.indexOf("boundary=");

        if (index==-1){

                return null;

        }

        String boundary =line.substring(index+9);
        boundary="--"+boundary;
```

96

```java
        return boundary;

}

private String[] extractDispositionInfo(String line) throws IOException {

        String[] retval=new String[3];
        String origline=line;
        line=origline.toLowerCase();

        int start =line.indexOf("content-disposition");
        int end =line.indexOf(";");
        if (start==-1||end==-1){

                throw new IOException("cont disposition corrupt" + origline);

        }

        String disposition=line.substring(start+21,end);
        if (!disposition.equals("form-data")){

                throw new IOException ("invalid content disposition"+disposition);

        }

        start=line.indexOf("name=\"",end);
        end=line.indexOf("\"",start+7);
        if (start==-1||end==-1){

                throw new IOException("content dispostion corrupt"+origline);

        }

        String name=origline.substring(start+6,end);
        String filename=null;
        start =line.indexOf("filename=\"",end+2);
        end=line.indexOf("\"",start+10);
        if (start!=-1 && end!=-1){

                filename=origline.substring(start+10,end);

                int slash=Math.max(filename.lastIndexOf('/'),filename.lastIndexOf('\\'));

                if (slash >-1){

                        filename=filename.substring(slash+1);

                }

                if (filename.equals("")) filename="unknown";

         }

         retval[0]=disposition;
         retval[1]=name;
         retval[2]=filename;

         return retval;

}


private String extractContentType(String line) throws IOException {

        String contentType=null;
        //convert the line to a lowercase string
        String origline = line;
```

97

```java
        line = origline.toLowerCase();

        if(line.startsWith("content-type")) {

                int start=line.indexOf(" ");

                if(start == -1) {

                        throw new IOException("content type corrupt: " + origline);

                }

                contentType=line.substring(start + 1);

        }
        else if(line.length() != 0) {

                //no content typt,so should be empty
                throw new IOException("malformed line after disposition:" + origline);

        }

        return contentType;


}
}




//A class to hold information about an uploaded file

class UploadedFile {

        private String dir;
        private String filename;
        private String type;


UploadedFile(String dir,String filename,String type) {

        this.dir=dir;
        this.filename=filename;
        this.type=type;

}

public String getContentType() {
return type;
}

public String getFilesystemName() {

        return filename;

}


public File getFile() {

        if(dir == null || filename == null) {

                return null;

        }
```

98

```
                else {

                        return new File(dir + File.separator + filename);

                }


        }

}



//A class to aid in reading multipart/form-data from a ServletInputStream
//It keeps track of how many bytes have been read & detects when the
//content-Length limit has been reached .this is necessary bcos some
//servlet engines are slow to notice the end of stream

class MultipartInputStreamHandler {

        ServletInputStream in;
        String boundary;
        int totalExpected;
        int totalRead=0;
        byte[] buf = new byte[8*1024];

public MultipartInputStreamHandler(ServletInputStream in, String boundary, int
totalExpected) {

        this.in=in;
        this.boundary=boundary;
        this.totalExpected=totalExpected;

}

public String readLine() throws IOException {

        StringBuffer sbuf=new StringBuffer();
        int result;
        String line;

        do {

                result=this.readLine(buf,0,buf.length);
                if (result != -1){

                        sbuf.append(new String(buf,0,result,"ISO-8859-1"));

                }

        }while(result == buf.length);

        if(sbuf.length() == 0 ) {

                return null;

        }

        sbuf.setLength(sbuf.length() - 2);
        return sbuf.toString();


}


public int readLine(byte b[],int off,int len) throws IOException {

        if(totalRead >= totalExpected) {
```

```java
                        return -1;

                }
                else {

                        int result=in.readLine(b,off,len);

                        if(result > 0) {

                                totalRead += result;

                        }

                        return result;

                }


}
}
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;


/**
 * This class takes information of the new project, connects to the database
 * and then if successful it creates the project folder within the file
 * system.
 * @see java.servlet.http
 * @author J. A. McDonald III
 */
public class processNewProject extends HttpServlet {

    String projName;
    String projVer;
    String userName;
    String firstName;
    String lastName;

    private String driverName = "org.gjt.mm.mysql.Driver";
    private String urlName = "jdbc:mysql://131.120.9.87:3306/capsdata";

    private String baseDirectory =  "/c/caps1/Webapps/tomcat/webapps/CapsWeb/capsusers";
    //private String baseDirectory = "D:\\program
files\\allaire\\jrun\\servers\\default\\CapsWeb\\capsusers";
    private Timestamp dateLastAccess;

    Vector projectNames;
    Vector projectVersions;

    private Driver       driver;

    private Connection   connection;

    private PrintWriter  out;

    private RequestDispatcher rd;

    HttpSession session;

    HttpServletRequest request;
    HttpServletResponse response;

/**
* Processes the user request and responds via an HTML output
```

```java
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
*/
public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException{

    res.setContentType("text/html");

    request = req;
    response = res;

    projName = request.getParameter("projName");
    projVer = request.getParameter("projVer");

    session = request.getSession();
    firstName = (String)session.getAttribute("firstname");
    lastName = (String)session.getAttribute("lastname");
    projectNames = (Vector)session.getAttribute( "projectNames" );
    projectVersions = (Vector)session.getAttribute( "projectVersions" );
    userName = (String)session.getAttribute( "username" );

    out = response.getWriter();

    topOfPage();

    if( checkProjectNames() ){

        if( createProject() ){

            out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
            "\"#0000FF\"><b>Project folders created successfully.</b></font></p>");
            resetNames();
            middleOfPage();
            bottomOfPage();

        }
        else{

            out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
            "\"#0000FF\"><b>There was a system problem.</b></font></p>");
            out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
            "\"#0000FF\"><b>Your project was not created.</b></font></p>");
            out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
            "\"#0000FF\"><b>Please Try Again.</b></font></p>");
            projectForm();
            bottomOfPage();


        }

    }
    else{

        out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
        "\"#0000FF\"><b>You already have a project with this name.</b></font></p>");
        out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
        "\"#0000FF\"><b>Please choose another project name.</b></font></p>");
        projectForm();
        bottomOfPage();


    }

    out.close();

}
```

```java
    /**
     * Establishes a connection to the database
     */
    private void establishConnection( ){

        Properties props = new Properties();

        props.put("user",     "mcdonald");
        props.put("password", "mcdonald");

        try {

            driver  = (Driver) Class.forName( driverName ).newInstance();
            connection = driver.connect( urlName, props );

        }
        catch ( Exception e ) {

            out.println("<p> Error: Cannot establish a connection to the database");
        }

    }
    /**
     *Checks to see if project name is unique
     *@return <code>boolean</code>
     */
    private boolean checkProjectNames(){

        String tempName = "";

        boolean isUnique = true;


        Enumeration e = projectNames.elements();

        while( e.hasMoreElements() ){

            tempName = (String)e.nextElement();

            if(tempName.equals(projName)){

                    return false;

            }

        }

        return true;


    }
    /**
     *Outputs an HTML Form for user entry of new project name
     */
    private void projectForm(){

        out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
        "\"#0000FF\"><form method ="+"\"get\" action ="+
        "\"/CapsWeb/servlet/processNewProject\"><font color=blue> New Project Name:<input
type=text size="+
        "\"25\" name ="+"\"projName\">");
        out.println("<form method ="+"\"get\" action ="+
        "\"/CapsWeb/servlet/processNewProject\"><font color=blue> New Project Version:<input
type=text size="+
        "\"4\" name ="+"\"projVer\"></font></p>");
        out.println("<center><input type="+"\"submit\" value="+
        "\"Create\">  <input type=reset></center>");

    }
```

```java
/**
 *Attempts to create a new project
 *@return <code>boolean</code>
 */
private boolean createProject(){

    if( updateDB() ){

        if( updateDir() ){

            return true;

        }
        else{

            rollBackDB();

            return false;

         }

        }
        else{

                return false;

        }


}
/**
 *Updates the project Folder in the file system
 *@return <code>boolean</code>
 */
private boolean updateDir(){

    String fileName = baseDirectory + File.separator + userName +
    File.separator + projName+ File.separator +projVer+ File.separator + "temp";

    try {

         File td = new File( fileName );
        td.mkdirs();
        return true;

    }
    catch (Exception e) {

         out.println(e);

         eturn false;

     }

}

/**
 *Attempts to update the database with the new project
 *@return <code>boolean</code>
 */
private boolean updateDB(){

   establishConnection();

   Statement sqlStmt = null;

   dateLastAccess = new Timestamp(System.currentTimeMillis());

   String inputtime = dateLastAccess.toString();
```

103

```java
        try {

            String query = "INSERT into project values ('" +projName+"','"+projVer+"','"+
                            userName+"','"+inputtime+"')";

            sqlStmt = connection.createStatement( );

            sqlStmt.executeUpdate( query );

            connection.close();

            sqlStmt.close();

            return true;

            }

            catch (Exception e){

                out.println(e);

                return false;

            }

}


private void rollBackDB(){

}
/**
 *Used to reset the vectors containing project names and version after new project
 * is added
 */
private void resetNames(){

    establishConnection();

    Statement sqlStmt = null;
    ResultSet rs      = null;

    String tempProjectName = "";
    String tempProjectVersion = "";

    try {

        String query = "select projectname, projectversion from project where username
="+
        "'"+userName+"' order by projectname";

        sqlStmt = connection.createStatement( );

        rs       = sqlStmt.executeQuery( query );

        projectNames.clear();
        projectVersions.clear();

        while ( rs.next() ){

            tempProjectName = rs.getString("projectname");
            projectNames.addElement(tempProjectName);
            tempProjectVersion = rs.getString("projectversion");
            projectVersions.addElement(tempProjectVersion);
```

104

```
            }

            //close connection to the database
            connection.close();
            sqlStmt.close();

            session.setAttribute("projectNames", projectNames );
            session.setAttribute("projectVersions", projectVersions);
            listProjects( projectNames, projectVersions );


        }
        catch (Exception e) {

            out.println(e);

        }


    }
    /**
     *Outputs a list of the projects and versions
     *@param project Vector
     *@param version Vector
     */
    private void listProjects( Vector project, Vector version ){

        String tempName = "";
        String tempVer  = "";

        Enumeration e = project.elements();
        Enumeration f = version.elements();

        out.println("<table border="+"\"0\"  align="+"\"center\" cellspacing="+
        "\"8\"  cellpadding="+"\"0\" >");

        out.println("<tr>");

        out.println("<td  align="+"\"left\" ><font size="+"\"3\" color="+
        "\"#0000FF\"><u>Project</font></u></td>");

        out.println("<td  align="+"\"left\" ><font size="+"\"3\" color="+
        "\"#0000FF\"><u>Version</font></u></td>");

        out.println("</tr>");

        while( e.hasMoreElements() && f.hasMoreElements() ){

            tempName = (String)e.nextElement();
            tempVer  = (String)f.nextElement();

            out.println("<tr>");

            out.println("<td  align="+"\"left\" ><font size="+"\"3\" color="+
            "\"#0000FF\">"+tempName+"</font></td>");

            out.println("<td  align="+"\"left\" ><font size="+"\"3\" color="+
            "\"#0000FF\">"+tempVer +"</font></td>");

            out.println("</tr>");


        }

        out.println("</table>");


    }
    /**
```

```java
 *Outputs the top of the HTML Page
 */
private void topOfPage(){

    out.println("<html><head><title>Distributed Computer Aided Prototyping
System</title>");
    out.println("<style fprolloverstyle>A:hover {color: #FF0000; font-size: 14pt;
vertical-align: 10; font-weight: bold}</style>");
    out.println("</head><hr color="+"\"#0000FF\">");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>Distributed Computer Aided Prototyping
System</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>Project Manager</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>User:  "+firstName+" "+lastName+"</i></b></font></p>");

    out.println("<hr color="+"\"#0000FF\"><br>");

    out.println("<body background=\"/CapsWeb/bg2.gif\">");

}
/**
 *Outputs the middle of the HTML Page
 */
private void middleOfPage(){

    out.println("<table border="+"\"0\"  align="+"\"center\" cellpadding="+"\"4\" >");

    out.println("<tr>");

    out.println("<td  align="+"\"center\" ><font size="+"\"4\" color="+
    "\"#0000FF\"><a href="+"\"/CapsWeb/servlet/createProject\">Create New
Project</a></font></td>");

    out.println("<td  align="+"\"center\" ><font size="+"\"4\" color="+
    "\"#0000FF\"><a href="+"\"/CapsWeb/servlet/createProjectVersion\">Create New Version
of a Project</a></font></td>");

    out.println("<td  align="+"\"center\" ><font size="+"\"4\" color="+
    "\"#0000FF\"><a href="+"\"/CapsWeb/servlet/setProject\">Set Project to Work
On</a></font></td>");

    out.println("</tr>");

    out.println("</table>");


}
/**
 *Outputs the bottom of the HTML Page
 */
private void bottomOfPage(){

    out.println("<br><br><br>"+"<hr color="+"\"#0000FF\">");
    out.println("<center><a href="+"\"/CapsWeb/servlet/logout\">Logout</a></center>");
    out.println("</html>");


}

}
```

106

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;


/**
 * This class takes information of the new project version, connects to the database
 * and then if successful it creates the project folder within the file
 * system.
 * @see java.servlet.http
 * @author J. A. McDonald III
 */
public class processNewVersion extends HttpServlet {

    String projName;
    String projVer;
    String userName;
    String firstName;
    String lastName;

    private String driverName = "org.gjt.mm.mysql.Driver";
    private String urlName = "jdbc:mysql://131.120.9.87:3306/capsdata";

    private String baseDirectory = "/c/caps1/Webapps/tomcat/webapps/CapsWeb/capsusers";
    //private String baseDirectory = "D:\\program
files\\allaire\\jrun\\servers\\default\\CapsWeb\\capsusers";


    private Timestamp dateLastAccess;

    Vector projectNames;
    Vector projectVersions;


    private Driver       driver;

    private Connection   connection;

    private PrintWriter  out;

    private RequestDispatcher rd;

    HttpSession session;

    HttpServletRequest request;
    HttpServletResponse response;

/**
* Processes the user request and responds via an HTML output
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
*/
public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException{

    res.setContentType("text/html");

    request = req;
    response = res;

    projName = request.getParameter("projName");
    projVer = request.getParameter("projVer");

    session = request.getSession();
    firstName = (String)session.getAttribute("firstname");
    lastName = (String)session.getAttribute("lastname");
    projectNames = (Vector)session.getAttribute( "projectNames" );
```
107

```java
        projectVersions = (Vector)session.getAttribute( "projectVersions" );
        userName = (String)session.getAttribute( "username" );
        out = response.getWriter();

        topOfPage();

        if( !checkProjectNames() ){

            if( createVersion() ){

                out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
                "\"#0000FF\"><b>New Version for "+ projName+
                " created successfully.</b></font></p>");
                resetNames();
                middleOfPage();
                bottomOfPage();

            }
            else{

                out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
                "\"#0000FF\"><b>There was a system problem.</b></font></p>");
                out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
                "\"#0000FF\"><b>Your project was not created.</b></font></p>");
                out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
                "\"#0000FF\"><b>Please Try Again.</b></font></p>");
                versionForm();
                bottomOfPage();


            }

        }
        else{

            out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
            "\"#0000FF\"><b>You don't have a project with this name.</b></font></p>");
            out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
            "\"#0000FF\"><b>Please resubmit the project name.</b></font></p>");
            listProjects( projectNames, projectVersions );
            versionForm();
            bottomOfPage();


        }
        out.close();

    }

    /**
     * Establishes a connection to the database
     */
    private void establishConnection( ){

        Properties props = new Properties();

        props.put("user",     "mcdonald");
        props.put("password", "mcdonald");
        try {

            driver    = (Driver) Class.forName( driverName ).newInstance();
            connection = driver.connect( urlName, props );

        }
        catch ( Exception e ) {

            out.println("<p> Error: Cannot establish a connection to the database");
        }
```

```java
}
/**
 *Checks the project names for uniqueness
 *@return <code>boolean</code>
 */
private boolean checkProjectNames(){

    String tempName = "";

    boolean isUnique = true;

    Enumeration e = projectNames.elements();

    while( e.hasMoreElements() ){

        tempName = (String)e.nextElement();

        if(tempName.equals(projName)){

                return false;

        }

    }

    return true;


}
/**
 *Outputs a form to enter the new project and new version
 */
private void versionForm(){

    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><form method ="+"\"get\" action ="+
    "\"/CapsWeb/servlet/processNewVersion\"><font color=blue> New Project Name:<input
type=text size="+
    "\"25\" name ="+"\"projName\">");
    out.println("<form method ="+"\"get\" action ="+
    "\"/CapsWeb/servlet/processNewVersion\"><font color=blue> New Project Version:<input
type=text size="+
    "\"4\" name ="+"\"projVer\"></font></p>");

    out.println("<center><input type="+"\"submit\" value="+
    "\"Create\">  <input type=reset></center>");

}
/**
 *Updates database and file system with new version
 *@return <code>boolean</code>
 */
private boolean createVersion(){

        if( updateDB() ){

                if( updateDir() ){

                        return true;

                }
                else{

                        rollBackDB();

                        return false;
```

109

```java
                }

            }
            else{

                    return false;

            }


}

/**
 *Updates the directory structor in the file system with the new project folder
 *@return <code>boolean</code>
 */
private boolean updateDir(){

    String fileName = baseDirectory + File.separator + userName +
    File.separator + projName+ File.separator +projVer+ File.separator + "temp";

   try {

       File td = new File( fileName );

       td.mkdirs();

        return true;

        }
     catch (Exception e) {

         out.println(e);

        return false;

    }

}
/**
 *Updates the database
 *@return <code>boolean</code>
 */
private boolean updateDB(){

    establishConnection();

    Statement sqlStmt = null;

    dateLastAccess = new Timestamp(System.currentTimeMillis());

    String inputtime = dateLastAccess.toString();

    try {

        String query = "INSERT into project values ('" +projName+"','"+projVer+"','"+
                        userName+"','"+inputtime+"')";

         sqlStmt = connection.createStatement( );

         sqlStmt.executeUpdate( query );

         connection.close();

         sqlStmt.close();

         return true;

    }
```

```java
      catch (Exception e){

          out.println(e);

          return false;

      }

}


private void rollBackDB(){

}

/**
 *Resets the Vector of project names after the new project is added
 */
private void resetNames(){

    establishConnection();

    Statement sqlStmt = null;
    ResultSet rs      = null;

    String tempProjectName = "";
    String tempProjectVersion = "";

    try {

        String query = "select projectname, projectversion from project where username
="+
        "'"+userName+"' order by projectname";
        sqlStmt = connection.createStatement( );

        rs      = sqlStmt.executeQuery( query );

        projectNames.clear();
        projectVersions.clear();

        while ( rs.next() ){

            tempProjectName = rs.getString("projectname");
            projectNames.addElement(tempProjectName);
            tempProjectVersion = rs.getString("projectversion");
            projectVersions.addElement(tempProjectVersion);


        }

        //close connection to the database
        connection.close();
        sqlStmt.close();

        session.setAttribute("projectNames", projectNames );
        session.setAttribute("projectVersions", projectVersions);
        listProjects( projectNames, projectVersions );




        }

        catch (Exception e) {

            out.println(e);
```

```java
        }

}

/**
 *Lists the projects names and version
 *@param project Vector
 *@param version Vector
 */
private void listProjects( Vector project, Vector version ){

    String tempName = "";
    String tempVer  = "";

    Enumeration e = project.elements();
    Enumeration f = version.elements();

    out.println("<table border="+"\"0\"  align="+"\"center\" cellspacing="+
    "\"8\"  cellpadding="+"\"0\" >");

    out.println("<tr>");

    out.println("<td  align="+"\"left\" ><font size="+"\"3\" color="+
    "\"#0000FF\"><u>Project</font></u></td>");

    out.println("<td  align="+"\"left\" ><font size="+"\"3\" color="+
    "\"#0000FF\"><u>Version</font></u></td>");

    out.println("</tr>");

    while( e.hasMoreElements() && f.hasMoreElements() ){

        tempName = (String)e.nextElement();
        tempVer  = (String)f.nextElement();

        out.println("<tr>");

        out.println("<td  align="+"\"left\" ><font size="+"\"3\" color="+
        "\"#0000FF\">"+tempName+"</font></td>");

        out.println("<td  align="+"\"left\" ><font size="+"\"3\" color="+
        "\"#0000FF\">"+tempVer +"</font></td>");

        out.println("</tr>");


    }
    out.println("</table>");


}

/**
 *Outputs the top of the HTML Page
 */
private void topOfPage(){

    out.println("<html><head><title>Distributed Computer Aided Prototyping
System</title>");
    out.println("<style fprolloverstyle>A:hover {color: #FF0000; font-size: 14pt;
vertical-align: 10; font-weight: bold}</style>");
    out.println("</head><hr color="+"\"#0000FF\">");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>Distributed Computer Aided Prototyping
System</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>Project Manager</i></b></font></p>");
```

112

```java
    out.println("<p align="+"\"center\">"+"<font size="+"\"5\" color="+
    "\"#0000FF\">"+"<b><i>User:  "+firstName+" "+lastName+"</i></b></font></p>");

    out.println("<hr color="+"\"#0000FF\">"+"<br>");

    out.println("<body background=\"/CapsWeb/bg2.gif\">");

}

/**
 *Outputs the middle portion of the HTML Page
 */
private void middleOfPage(){

    out.println("<table border="+"\"0\"  align="+"\"center\" cellpadding="+
    "\"4\" >");

    out.println("<tr>");

    out.println("<td  align="+"\"center\" >"+"<font size="+"\"4\" color="+
    "\"#0000FF\">"+"<a href="+
    "\"/CapsWeb/servlet/createProject\">Create New Project</a></font></td>");

    out.println("<td  align="+"\"center\" >"+"<font size="+"\"4\" color="+
    "\"#0000FF\">"+"<a href="+
    "\"/CapsWeb/servlet/createProjectVersion\">Create New Version of a
Project</a></font></td>");

    out.println("<td  align="+"\"center\" >"+"<font size="+"\"4\" color="+
    "\"#0000FF\">"+"<a href="+
    "\"/CapsWeb/servlet/setProject\">Set Project to Work On</a></font></td>");

    out.println("</tr>");

    out.println("</table>");

}
/**
 *Outputs the bottom portion of the HTML Page
 */
private void bottomOfPage(){

    out.println("<br><br><br>"+"<hr color="+"\"#0000FF\">");
    out.println("<center><a href="+"\"/CapsWeb/servlet/logout\">Logout</a></center>");
    out.println("</html>");


}



    }
```

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;

/**
 * This class creates an HTML output allowing users to choose b/n options
 * allowing them to develop the prototypes such as launching the editor
 * translating, scheduler, compiling, and execution as well as changing the project
 * to be worked on.  This is the first of two managers
 * @see manager.class
 * @see java.servlet.http
 * @author J. A. McDonald III
 */
public class projectManager extends HttpServlet {

    private String userUserName;

    private String firstName;
    private String lastName;


    private String driverName = "org.gjt.mm.mysql.Driver";
    private String urlName = "jdbc:mysql://131.120.9.87:3306/capsdata";



    private Driver        driver;

    private Connection    connection;

    private PrintWriter   out;

    private RequestDispatcher rd;

    HttpSession session;

    HttpServletRequest request;
    HttpServletResponse response;

    boolean dbStatus = false;
/**
* Processes the user request and responds via an HTML output
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
*/
public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException{

    res.setContentType("text/html");

    request = req;
    response = res;
    session = req.getSession();
    userUserName = (String)session.getAttribute("username");
    firstName = (String)session.getAttribute("firstname");
    lastName = (String)session.getAttribute("lastname");
    out = res.getWriter();

    topOfPage();

    retrieveProjects( req, res );

    middleOfPage();

    bottomOfPage();

    out.close();
```

```java
}

   /**
    * Establishes a connection to the database
     *@return <code>boolean</code> true if good otherwise false
    */
private boolean establishConnection( ){

    Properties props = new Properties();

    props.put("user",     "mcdonald");
    props.put("password", "mcdonald");

    try {

        driver   = (Driver) Class.forName( driverName ).newInstance();
        connection = driver.connect( urlName, props );
        return true;

    }
    catch ( Exception e ) {

        return false;

    }
}
/**
 *Outputs the top of the HTML Page
 */
private void topOfPage(){

    out.println("<html><head><title>Distributed Computer Aided Prototyping
System</title>");
    out.println("<style fprolloverstyle>A:hover {color: #FF0000; font-size: 14pt;
vertical-align: 10; font-weight: bold}</style>");
    out.println("</head><hr color="+"\"#0000FF\">");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>Distributed Computer Aided Prototyping
System</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>Project Manager</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>User:  "+firstName+" "+lastName+"</i></b></font></p>");

    out.println("<hr color="+"\"#0000FF\"><br>");

    out.println("<body background=\"/CapsWeb/bg2.gif\">");

}

/**
 *Outputs the middle of the HTML Page
 */
private void middleOfPage(){

    out.println("<table border="+"\"0\"  align="+"\"center\" cellpadding="+"\"4\" >");

    out.println("<tr>");

    out.println("<td  align="+"\"center\" ><font size="+"\"4\" color="+
    "\"#0000FF\"><a href="+
    "\"/CapsWeb/servlet/createProject\">Create New Project</a></font></td>");

    out.println("<td  align="+"\"center\" ><font size="+"\"4\" color="+
    "\"#0000FF\"><a href="+
    "\"/CapsWeb/servlet/createProjectVersion\">Create New Version of a
Project</a></font></td>");

    out.println("<td  align="+"\"center\" ><font size="+"\"4\" color="+
```

115

```java
            "\"#0000FF\"><a href="+
            "\"/CapsWeb/servlet/setProject\">Set Project to Work On</a></font></td>");

        out.println("</tr>");

        out.println("</table>");

    }
    /**
     *Outputs the bottom of the HTML Page
     */
    private void bottomOfPage(){

        out.println("<br><br><br>"+"<hr color="+"\"#0000FF\">");
        out.println("<center><a href="+"\"/CapsWeb/servlet/logout\">Logout</a></center>");

        out.println("</html>");


    }

    /**
     *Retrieves the projects from retrieveProjects Servlets
     *@param HttpServletRequest req
     *@param HttpServletResponse res
     */
    private void retrieveProjects(HttpServletRequest req, HttpServletResponse res)throws
    IOException{

        rd = getServletContext().getRequestDispatcher("/servlet/retrieveProjects");

        try{

            rd.include( req, res );

        }
        catch( IOException e ){

            out.println("ProblemIO");

        }
        catch( ServletException e ){

            out.println("ProblemServlet");
            out.println(e);

        }

    }


    }
```

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;


/**
 * This class is used to retrieve project names and version numbers and places
 * them into two separate vectors and makes them available via the http session
 * @see java.servlet.http
 * @author J. A. McDonald III
 */
public class retrieveProjects extends HttpServlet {

    private String projectName;
    private String projectVersion;

    private String userUserName;
    private String timeStamp;

    private String driverName = "org.gjt.mm.mysql.Driver";
    private String urlName = "jdbc:mysql://131.120.9.87:3306/capsdata";

    private Driver       driver;

    private Connection   connection;

    private PrintWriter  out;

    HttpSession session;

    HttpServletRequest request;
    HttpServletResponse response;

/**
* Processes the user request and responds via an HTML output
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
*/
public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException{

    res.setContentType("text/html");

    request = req;
    response = res;

    session = req.getSession();
    userUserName = (String)session.getAttribute("username");

  out = res.getWriter();

  getProjects();

      //out.close();

}

  /**
  * Establishes a connection to the database
  */

private void establishConnection( ){

    Properties props = new Properties();

    props.put("user",     "mcdonald");
    props.put("password", "mcdonald");
```

117

```
        try {

                driver   = (Driver) Class.forName( driverName ).newInstance();
                connection = driver.connect( urlName, props );

        }
        catch ( Exception e ) {

                out.println("<p> Error: Cannot establish a connection to the database");
        }

}

/**
 *retrieves the projects and versions from the database and puts them in
 *Vectors
 */
private void getProjects(){

        int count = 0;
        establishConnection();

        Statement sqlStmt = null;
        ResultSet rs      = null;
        Vector projectNames = new Vector();
        Vector projectVersions = new Vector();
        String tempProjectName = "";
        String tempProjectVersion = "";

        try {

                String query = "select projectname, projectversion from project where username
="+
                "'"+userUserName+"' order by projectname";

                sqlStmt = connection.createStatement( );

                rs      = sqlStmt.executeQuery( query );

                while ( rs.next() ){

                        tempProjectName = rs.getString("projectname");
                        projectNames.addElement(tempProjectName);
                        tempProjectVersion = rs.getString("projectversion");
                        projectVersions.addElement(tempProjectVersion);
                        count++;

                }

                //close connection to the database
                connection.close();
                sqlStmt.close();

                if( count==0 ){

                        session.setAttribute("projectNames", projectNames );
                        session.setAttribute("projectVersions", projectVersions);
                        out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
                        "\"#0000FF\"><b><i>Currently you have no projects.</i></b></font></p>");

                }
                else{

                        session.setAttribute("projectNames", projectNames );
                        session.setAttribute("projectVersions", projectVersions);
                        listProjects( projectNames, projectVersions );
```

118

```java
            }


    }

    catch (Exception e) {

        out.println(e);

    }


}


/**
 *Outputs the project names and version in an HTML Form
 *@param project Vector
 *@param version Vector
 */
private void listProjects( Vector project, Vector version ){

    String tempName = "";
    String tempVer  = "";

    Enumeration e = project.elements();
    Enumeration f = version.elements();

    out.println("<table border="+"\"0\"  align="+"\"center\" cellspacing="+
    "\"8\"  cellpadding="+"\"0\" >");

    out.println("<tr>");

    out.println("<td  align="+"\"left\" ><font size="+"\"3\" color="+
    "\"#0000FF\"><u>Project</font></u></td>");

    out.println("<td  align="+"\"left\" ><font size="+"\"3\" color="+
    "\"#0000FF\"><u>Version</font></u></td>");

    out.println("</tr>");

    while( e.hasMoreElements() && f.hasMoreElements() ){

        tempName = (String)e.nextElement();
        tempVer  = (String)f.nextElement();

        out.println("<tr>");

        out.println("<td  align="+"\"left\" ><font size="+"\"3\"
color="+"\"#0000FF\">"+tempName+"</font></td>");

        out.println("<td  align="+"\"left\" ><font size="+"\"3\"
color="+"\"#0000FF\">"+tempVer +"</font></td>");

        out.println("</tr>");


    }

     out.println("</table>");

}


}
```

119

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;

/**
 * This class invokes the CAPS scheduler
 * @see java.servlet.http
 * @author J. A. McDonald III
 */
public class scheduler extends HttpServlet {

    String selectedProject = "";
    String selectedVersion = "";
    String userUserName = "";
    private String firstName;
    private String lastName;

    private String baseDirectory = "/c/caps1/Webapps/tomcat/webapps/CapsWeb/capsusers/";

    StringBuffer all;


    private PrintWriter  out;

    HttpSession session;

    HttpServletRequest request;
    HttpServletResponse response;


/**
 * Processes the user request and responds via an HTML output
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
 */
public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException{

    res.setContentType("text/html");

    request = req;
    response = res;
    session=request.getSession();

    selectedProject = (String)session.getAttribute("selectedProject");
    selectedVersion = (String)session.getAttribute("selectedVersion");
    userUserName = (String)session.getAttribute("username");
    firstName = (String)session.getAttribute("firstname");
    lastName = (String)session.getAttribute("lastname");

    out = response.getWriter();

    topOfPage();

    schedulePrototype();

    middleOfPage( all );

    bottomOfPage();

    out.close();

}//end of doGet
/**
 *Invokes the CAPS Scheduler
 */
public void schedulePrototype(){
```

```java
    String command = "/c/caps1/Webapps/tomcat/webapps/CapsWeb/bin/make.script" +
    " " + userUserName + " " + selectedProject + " " + selectedVersion;

    System.out.println(command);
        //out.println(command);

    try{

        Runtime run = Runtime.getRuntime();
        run.exec(command);

     }
     catch(IOException ex ){

         out.println(ex);


     }

}//end schedulePrototype()

/**
 *Reads the log file of the scheduler
 */
public void readLogFile(){

    File file = new File( baseDirectory + userUserName + "/" + selectedProject +
                "/" + selectedVersion + "/temp/" + selectedProject + ".log");
    String part;

    all = new StringBuffer("");

    if(file != null){

        try{
            FileReader fr = new FileReader(file);
            BufferedReader br= new BufferedReader(fr);
            while( (part=br.readLine()) != null){
                all.append(part);
                all.append('\n');
            }
            br.close();
            fr.close();

            }
            catch( IOException fnf ){

                out.println( fnf );

             }

        }
        else{

            out.println( "File not found.");

        }

}//end of readLogFile()

/**
 *Outputs the top of the HTML Page
 */
private void topOfPage(){

    out.println("<html><head><title>Distributed Computer Aided Prototyping
System</title>");
    out.println("<style fprolloverstyle>A:hover {color: #FF0000; font-size: 14pt;
vertical-align: 10; font-weight: bold}</style>");
    out.println("</head><hr color="+"\"#0000FF\">");
```
121

```
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>Distributed Computer Aided Prototyping
System</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>Web-CAPS Scheduler</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
    "\"#0000FF\"><b><i>User:  "+firstName+" "+lastName+"</i></b></font></p>");
    out.println("<hr color="+"\"#0000FF\"><br>");
    out.println("<body background=\"/CapsWeb/bg2.gif\">");

}

/**
 *Outputs the middle of the HTML Page
 */
private void middleOfPage( StringBuffer all ){

    out.println("<table border="+"\"0\"  align="+"\"center\" cellpadding="+
    "\"4\" >");

    out.println("<tr>");

    out.println("<td  align="+"\"center\" ><font size="+"\"4\" color="+
    "\"#0000FF\"><a href="+"\"/CapsWeb/capsusers/"+userUserName+"/"+
    selectedProject+"/"+selectedVersion+"/temp/"+ selectedProject +
    ".diag\" target=\"Resource Window\">View Schedule</a></font></td>");

    out.println("<td  align="+"\"center\" ><font size="+"\"4\" color="+
    "\"#0000FF\"><a href="+"\"/CapsWeb/capsusers/"+userUserName+"/"+selectedProject+
    "/"+selectedVersion+
    "/temp/schedule.err\" target=\"Resource Window\">View Scheduling
Errors</a></font></td>");
    out.println("<td  align="+"\"center\" ><font size="+"\"4\" color="+
    "\"#0000FF\"><a href="+"\"/CapsWeb/capsusers/"+userUserName+"/"+
    selectedProject+"/"+selectedVersion+
    "/temp/schedule.trace\" target=\"Resource Window\">View Scheduling
Log</a></font></td>");

    out.println("<td  align="+"\"center\" ><font size="+"\"4\" color="+
    "\"#0000FF\"><a href="+"\"/CapsWeb/servlet/manager\">Return to
Manager</a></font></td>");

    out.println("</tr>");

    out.println("</table>");



}
/**
 *Outputs the bottom of the HTML Page
 */
private void bottomOfPage(){

    out.println("<br><br><br>"+"<hr color="+"\"#0000FF\">");
    out.println("<center><a href="+"\"/CapsWeb/servlet/logout\">Logout</a></center>");
    out.println("</html>");


}


}
```

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;

/**
 *
 *
 *
 *
 *
 *
 */
public class selectProject extends HttpServlet {

    String selectedVersion = null;
    String selectedProject = null;
    String username = null;


    private Driver       driver;

    private Connection   connection;

    private PrintWriter  out;



    HttpSession session;

    HttpServletRequest request;
    HttpServletResponse response;


    public void doGet(HttpServletRequest req, HttpServletResponse res)
    throws IOException{

        // Sets the content type to html text


        res.setContentType("text/html");

        request = req;
        response = res;

        selectedVersion = request.getParameter("selectedVersion");
        // Gets the PrintWriter object for sending HTML commands

        out = response.getWriter();


        session = req.getSession();
        session.setAttribute( "selectedVersion", selectedVersion );
        selectedProject = (String)session.getAttribute("selectedProject");
        username = (String)session.getAttribute("username");

        topOfPage();

        out.println("<p align="+"\"center\"><font size="+"\"3\"
color="+"\"#0000FF\"><b>User--"+username+"  Project--
"+selectedProject+"  Version--"+selectedVersion+".</b></font></p>");

        out.println("<center><font size="+"\"4\" color="+"\"#0000FF\"><b><a
href="+"\"/CapsWeb/servlet/capsjnlp\">Launch Editor</a>  ");

        out.println("<a href="+"\"/CapsWeb/servlet/upLoad\">Upload Project
Files</a>  ");
```

123

```
        out.println("<a
href="+"\"/CapsWeb/servlet/translate\">Translate</a>  ");

        out.println("<a
href="+"\"/CapsWeb/servlet/scheduler\">Schedule</a>  ");

        out.println("<a href="+"\"/CapsWeb/servlet/compile\">Compile</a>  ");

        out.println("<a
href="+"\"/CapsWeb/servlet/executejnlp\">Execute</a>  ");

        out.println("<a href="+"\"/CapsWeb/servlet/setProject\">Reset
Project</a></b></font></center>");

        bottomOfPage();

        out.close();


    }


private void topOfPage(){

        out.println("<html><head><title>Distributed Computer Aided Prototyping
System</title>");
        out.println("<style fprolloverstyle>A:hover {color: #FF0000; font-size: 14pt;
vertical-align: 10; font-weight: bold}</style>");
        out.println("</head><hr color="+"\"#0000FF\">");
        out.println("<p align="+"\"center\"><font size="+"\"5\"
color="+"\"#0000FF\"><b><i>Distributed Computer Aided Prototyping
System</i></b></font></p>");
        out.println("<p align="+"\"center\"><font size="+"\"5\"
color="+"\"#0000FF\"><b><i>Project Manager</i></b></font></p>");

        out.println("<hr color="+"\"#0000FF\"><br>");

        out.println("<body background=\"/CapsWeb/bg2.gif\">");

}




private void bottomOfPage(){

        out.println("<br><br><br>"+"<hr color="+"\"#0000FF\">");


        out.println("<center><a
href="+"\"/CapsWeb/servlet/logout\">Logout</a></center>");


        out.println("</html>");


}



}
```

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;

/**
 * This class creates an HTML form where the user is allowed to choose a project
 * to work on
 * @see java.servlet.http
 * @author J. A. McDonald III
 */
public class setProject extends HttpServlet {

    private Driver       driver;

    private Connection   connection;

    private PrintWriter  out;

    Vector projectNames;

    Vector projectVersions;


    HttpSession session;

    HttpServletRequest request;
    HttpServletResponse response;

    boolean dbStatus = false;
/**
 * Processes the user request and responds via an HTML output
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
 */
public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException{

    res.setContentType("text/html");

    request = req;
    response = res;
    out = res.getWriter();
    session = req.getSession();
    projectNames = (Vector)session.getAttribute("projectNames");
    projectVersions = (Vector)session.getAttribute("projectVersions");

    topOfPage();

    listProjects( projectNames, projectVersions );

    bottomOfPage();

    out.close();

}

/**
 * Outputs the list of Projects and Versions
 *@param project Vector
 *@param version Vector
 */
private void listProjects( Vector project, Vector version ){

    int j = 0;

    String tempName = "";
```
125

```java
    String tempVer  = "";

    Enumeration e = project.elements();
    Enumeration f = version.elements();
    out.println("<form method="+"\"get\" action ="+"\"/CapsWeb/servlet/setVersion\"><font
color="+"\"#0000FF\"><center><h2>Choose a Project for your session.</h2>");
    out.println("<select name="+"\"selectedProject\">");

    while( e.hasMoreElements() && f.hasMoreElements() ){

        tempName = (String)e.nextElement();
        tempVer  = (String)f.nextElement();

        out.println("<option >"+tempName+"</option>");

        j++;

    }

     out.println("</select><br><br>");

    out.println("<center><input type="+"\"submit\" value="+"\"Set
Project\">  <input type=reset></center></form>");

    if( j == 0 ){

        out.println("<p align="+"\"center\"><font size="+"\"5\"
color="+"\"#0000FF\"><b>Currently you have no Projects in the System.</b></font></p>");

    }

}
/**
 *Outputs the top of the HTML Page
 */
private void topOfPage(){

    out.println("<html><head><title>Distributed Computer Aided Prototyping
System</title>");
    out.println("<style fprolloverstyle>A:hover {color: #FF0000; font-size: 14pt;
vertical-align: 10; font-weight: bold}</style>");
    out.println("</head><hr color="+"\"#0000FF\">");
    out.println("<p align="+"\"center\"><font size="+"\"5\"
color="+"\"#0000FF\"><b><i>Distributed Computer Aided Prototyping
System</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"5\"
color="+"\"#0000FF\"><b><i>Set Project</i></b></font></p>");
    out.println("<hr color="+"\"#0000FF\"><br>");
    out.println("<body background=\"/CapsWeb/bg2.gif\">");

}

/**
 *Outputs the bottom of the HTML Page
 */
private void bottomOfPage(){

    out.println("<br><br><br>"+"<hr color="+"\"#0000FF\">");
    out.println("<center><a href="+"\"/CapsWeb/servlet/logout\">Logout</a></center>");
    out.println("</html>");


}

}
```

126

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;

/**
 * This class creates an HTML form allowing the user to choose a version of the
 * project to work on
 * @see java.servlet.http
 * @author J. A. McDonald III
 */
public class setVersion extends HttpServlet {


    private Driver        driver;

    private Connection    connection;

    private PrintWriter  out;

    Vector projectNames;

    Vector projectVersions;

    String selectedProject = "";


    HttpSession session;

    HttpServletRequest request;
    HttpServletResponse response;

    boolean dbStatus = false;
/**
* Processes the user request and responds via an HTML output
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
*/
public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException{

    res.setContentType("text/html");

    request = req;
    response = res;

    out = res.getWriter();
    selectedProject = request.getParameter("selectedProject");

    session = req.getSession();
    session.setAttribute("selectedProject",selectedProject);

    projectNames = (Vector)session.getAttribute("projectNames");
    projectVersions = (Vector)session.getAttribute("projectVersions");

    topOfPage();

    listProjects( projectNames, projectVersions );

    bottomOfPage();

    out.close();

}

  /**
   * Displays a list of projects and versions
   *@param project Vector
   *@param version Vector
```

```java
    */
 private void listProjects( Vector project, Vector version ){

      int j = 0;

      String tempName = "";
      String tempVer  = "";

      Enumeration e = project.elements();
      Enumeration f = version.elements();

      out.println("<p align="+"\"center\"><font size="+"\"5\"
color="+"\"#0000FF\"><b><i>The Project set is:  "+selectedProject+"</i></b></font></p>");
      out.println("<form method="+"\"get\" action
="+"\"/CapsWeb/servlet/selectProject\"><font color=blue><center><h2>Choose a Version of
the Project.</h2>");
      out.println("<select name="+"\"selectedVersion\">");

      while( e.hasMoreElements() && f.hasMoreElements() ){

          tempName = (String)e.nextElement();
          tempVer  = (String)f.nextElement();

          if(tempName.equals(selectedProject)){

              out.println("<option >"+tempVer+"</option>");
          }
          j++;


      }

       out.println("</select><br><br>");

      out.println("<center><input type="+"\"submit\" value="+"\"Set
Version\">  <input type=reset></center></form>");


      if( j == 0 ){

          out.println("<p align="+"\"center\"><font size="+"\"5\"
color="+"\"#0000FF\"><b>Currently you have no Projects in the System.</b></font></p>");

      }


}
/**
 *Outputs the top of the HTML Page
 */
private void topOfPage(){

    out.println("<html><head><title>Distributed Computer Aided Prototyping
System</title>");
    out.println("<style fprolloverstyle>A:hover {color: #FF0000; font-size: 14pt;
vertical-align: 10; font-weight: bold}</style>");
    out.println("</head><hr color="+"\"#0000FF\">");
    out.println("<p align="+"\"center\"><font size="+"\"5\"
color="+"\"#0000FF\"><b><i>Distributed Computer Aided Prototyping
System</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"5\"
color="+"\"#0000FF\"><b><i>Set Project</i></b></font></p>");
    out.println("<hr color="+"\"#0000FF\"><br>");
    out.println("<body background=\"/CapsWeb/bg2.gif\">");

}

/**
 *Outputs the bottom of the HTML Page
 */
```

128

```
private void bottomOfPage(){

    out.println("<br><br><br>"+"<hr color="+"\"#0000FF\">");
    out.println("<center><a href="+"\"/CapsWeb/servlet/logout\">Logout</a></center>");
    out.println("</html>");

}


}
```

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;

/**
 * This class is used to pass down the file information to the CAPS Editor on
 * the client system.
 * @see java.servlet.http
 * @author J. A. McDonald III
 */
public class testDownload extends HttpServlet {

    private String driverName = "org.gjt.mm.mysql.Driver";
    private String urlName = "jdbc:mysql://131.120.9.87:3306/capsdata";


    private Driver        driver;

    private Connection    connection;

    private OutputStream  out;

    String userFolderNeeded = "";

    private String baseDirectory = "/c/caps1/Webapps/tomcat/webapps/CapsWeb/capsusers";

    //private String baseDirectory = "D:\\program
files\\allaire\\jrun\\servers\\default\\CapsWeb\\capsusers";



    Vector prototypeNames;

    HttpServletRequest request;
    HttpServletResponse response;

    String filesInFolder = "";
/**
* Processes the user request and responds via an HTML output
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
*/
public void doPost(HttpServletRequest req, HttpServletResponse res)throws IOException{

    res.setContentType("content/unknown");
    request = req;
    response = res;

    userFolderNeeded = request.getParameter("userFolder");

    getFile();


}
/**
 *gets the file for download
 */
public void getFile(){

    String filesToGet = baseDirectory + userFolderNeeded;
    File protoDir = new File (filesToGet);
    prototypeNames = new Vector (0, 2);
    File [] dirs = protoDir.listFiles ();
    String protoName = "";
```

130

```java
        for (int ix = 0; ix < dirs.length; ix++){

            protoName = dirs [ix].getName ();
            prototypeNames.addElement (protoName);
        }
        try{

            out = response.getOutputStream();

            ObjectOutputStream p = new ObjectOutputStream(out);

            p.writeObject(dirs);

        }
        catch(IOException e){



        }
        for (Enumeration f = prototypeNames.elements() ; f.hasMoreElements() ;) {

            String tempProto = "";

            tempProto = (String)f.nextElement();

                //out.write(tempProto);

        }


}
/**
* Processes the user request and responds via an HTML output
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
*/
public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException{

        doPost(req,res);

}


}
```

```java
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
import java.util.*;

/**
 * This class creates an HTML output allowing users to identify and upload
 * files to the file system
 * @see java.servlet.http
 * @author J. A. McDonald III
 */
public class upLoad extends HttpServlet {


    String selectedVersion = null;
    String selectedProject = null;
    String username = null;

    private Driver        driver;

    private Connection    connection;

    private PrintWriter  out;

    HttpSession session;

    HttpServletRequest request;
    HttpServletResponse response;


/**
 * Processes the user request and responds via an HTML output
 *@param HttpServletRequest req
 *@param HttpServletResponse res
 *@throws IOException
 */
public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException{

    // Sets the content type to html text
    res.setContentType("text/html");

    request = req;
    response = res;
    // Gets the PrintWriter object for sending HTML commands

    out = response.getWriter();
    session = req.getSession();

    selectedProject = (String)session.getAttribute("selectedProject");
    selectedVersion = (String)session.getAttribute("selectedVersion");
    username = (String)session.getAttribute("username");

    topOfPage();

    out.println("<p align="+"\"center\"><font size="+"\"3\" color="+
        "\"#0000FF\"><b>User--"+username+"  Project--"+
        selectedProject+"  Version--"+selectedVersion+
        ".</b></font></p>");
    formOutput( );

    bottomOfPage();

    out.close();


}//end of doGet

/**
 *Outputs the top of the HTML Page
```

```
*/
private void topOfPage(){

    out.println("<html><head><title>Distributed Computer Aided Prototyping
System</title>");
    out.println("<style fprolloverstyle>A:hover {color: #FF0000; font-size: 14pt;
vertical-align: 10; font-weight: bold}</style>");
    changeFileNumber();
    out.println("</head><hr color="+"\"#0000FF\">");
    out.println("<p align="+"\"center\"><font size="+
        "\"5\" color="+"\"#0000FF\"><b><i>Distributed Computer Aided Prototyping
System</i></b></font></p>");
    out.println("<p align="+"\"center\"><font size="+"\"5\" color="+
        "\"#0000FF\"><b><i>Upload Project Files</i></b></font></p>");
    out.println("<hr color="+"\"#0000FF\"><br>");
    out.println("<body background=\"/CapsWeb/bg2.gif\">");

}//end of topOfPage


/**
*  Outputs an HTML form for user input
*/

private void formOutput(){

    out.println("<center><font size="+"\"4\" color="+"\"#0000FF\"><form name=counter>");
    out.println("Number of Files to Upload:");
    out.println("<input type=text name=number size=5>");
    out.println("<input type=button value=\"Update\"
onClick=\"createForm(counter.number.value);\">");
    out.println("</form>");
    out.println("<br>");
    out.println("<FORM ACTION=\"/CapsWeb/servlet/UploadTest\" method=post
ENCTYPE=\"multipart/form-data\">");
    out.println("<!-- Placeholder for dynamic form contents -->");
    out.println("<span id=cust style=\"position:relative;\"></span>");
    out.println("<INPUT TYPE=SUBMIT><INPUT TYPE=RESET>");
    out.println("</FORM></font></center>");

}//end of formOutput


/**
*  Outputs Javascript to be used for client side processing
*/
private void changeFileNumber(){

    out.println("<SCRIPT LANGUAGE=\"JavaScript\">");
    out.println("<!-- Original:  Peter Hermus -->");
    out.println("<!-- Web Site:  http://come.to/speedpete -->");
    out.println("<!-- Begin");
    out.println("function createForm(number) {");
    out.println("data = \" \";");
    out.println("inter = \"'\";");
    out.println("if (number < 26 && number > -1) {");
    out.println("for (i=1; i <= number; i++) {");
    out.println("if (i < 10) spaces=\"     \";");
    out.println("else spaces=\"    \";");
    out.println("data = data + \"File Number \" + i + \" :\" + spaces");
    out.println("+ \"<input type='FILE' size=50 name=\" + inter");
    out.println("+ \"file\" + i + inter + \"'><br>\";");
    out.println("}");
    out.println("if (document.layers) {");
    out.println("document.layers.cust.document.write(data);");
    out.println("document.layers.cust.document.close();");
    out.println("}");
    out.println("else {");
    out.println("if (document.all) {");
    out.println("cust.innerHTML = data;");
    out.println(" }");
```

133

```
    out.println("    }");
    out.println("}");
    out.println("else {");
    out.println("window.alert(\"Please select up to 25 entries.\");");
    out.println("    }");
    out.println("}");
    out.println("//  End -->");
    out.println("</script>");


}//end of change file number

/**
* Outputs the bottom of the HTML Page
*/
private void bottomOfPage(){

    out.println("<br><br><br>"+"<hr color="+"\"#0000FF\">");
    out.println("<center><a href="+"\"/CapsWeb/servlet/logout\">Logout</a></center>");
    out.println("</html>");

}//end of bottomOfPage




}
```

```
import java.io.*;
import java.util.*;
import javax.servlet.http.*;
import javax.servlet.*;
import MultipartRequest;
/**
 * This class creates an HTML output allowing users to identify and upload
 * files to the file system
 * @see java.servlet.http
 * @author J. A. McDonald III
 */
public class UploadTest extends HttpServlet{

    String selectedVersion = null;
    String selectedProject = null;
    String username = null;
    String userFolder = null;
    //String baseFolder = "d:\\program
files\\allaire\\jrun\\servers\\default\\capsweb\\capsusers\\";
    private String baseFolder = "/c/caps1/Webapps/tomcat/webapps/CapsWeb/capsusers/";

    private PrintWriter  out;

    private RequestDispatcher rd;

    HttpSession session;

    public void doPost(HttpServletRequest req, HttpServletResponse res) throws
IOException,ServletException{

        res.setContentType("text/html");

        out = res.getWriter();

        session = req.getSession();

        username = (String)session.getAttribute("username");
        selectedProject = (String)session.getAttribute("selectedProject");
        selectedVersion = (String)session.getAttribute("selectedVersion");

        userFolder = username + File.separator + selectedProject + File.separator +
selectedVersion + File.separator;


        try {


                MultipartRequest multi = new MultipartRequest(req, baseFolder +
userFolder, 5*1024*1024 );

                topOfPage();
                out.println("<p align="+"\"center\"><font size="+"\"3\"
color="+"\"#0000FF\"><b>User--"+username+"  Project--
"+selectedProject+"  Version--"+selectedVersion+".</b></font></p>");

                //out.println("<html>");
                //out.println("<head> <title> UploadTest </title></head>");
                //out.println("<body>");
                //out.println("<h1>UploadTest</h1>");
                //out.println("<h3>Params:</h3>");
                //out.println("<pre>");

                //String temp =req.getParameter("first");
                //out.println(temp);

                //Enumeration params = multi.getParameterNames();

                //while (params.hasMoreElements()) {
                        //String name = (String)params.nextElement();
                        //String value = multi.getParameter(name);
```
135

```
            //out.println(name+" = "+value);
            //}

            //out.println("</pre>");

            //out.println("<p align="+\"center\"><font size="+\"5\"
color="+\"#0000FF\"><b><i>Uploaded Project Files</i></b></font></p>");

            Enumeration files = multi.getFileNames();

            out.println("<table border="+\"1\"  align="+\"center\"
cellspacing="+\"4\"  cellpadding="+\"2\" >");

             out.println("<tr>");

             out.println("<td  align="+\"center\" colspan=\"3\"><font size="+\"3\"
color="+\"#0000FF\"><u>Uploaded Project Files</font></u></td>");

             out.println("</tr>");

            while(files.hasMoreElements()) {

                    String name = (String)files.nextElement();

                    String filename = multi.getFilesystemName(name);

                    String type = multi.getContentType(name);

                    File f = multi.getFile(name);

                    out.println("<tr>");

                    out.println("<td  align="+\"left\" ><font size="+\"3\"
color="+\"#0000FF\">Filename:  "+filename+"</i></b></font></td>");

                     out.println("<td  align="+\"left\" ><font size="+\"3\"
color="+\"#0000FF\">Type:  "+type+"</i></b></font></td>");

                    if(f != null) {

                            out.println("<td  align="+\"left\" ><font size="+\"3\"
color="+\"#0000FF\">Length:  "+f.length()+"</i></b></font><td>");


//out.println("  Length:  "+f.length()+"</i></b></font></td>");

                            out.println();

                    }

                     out.println("</tr>");

            }

            out.println("</table>");
            out.println("<br>");
            out.println("<br>");

    }
    catch(Exception e) {

             e.printStackTrace(out);

    }

    //out.println("</body></html>");
```

136

```java
        out.println("<center><a href="+"\"/CapsWeb/servlet/manager\">Return to Project
Manager</a></center>");

        bottomOfPage();

}

public void doGet(HttpServletRequest req, HttpServletResponse res) throws
IOException,ServletException{

        doPost(req,res);

}


private void topOfPage(){

        out.println("<html><head><title>Distributed Computer Aided Prototyping
System</title>");
        out.println("<style fprolloverstyle>A:hover {color: #FF0000; font-size: 14pt;
vertical-align: 10; font-weight: bold}</style>");
        out.println("</head><hr color="+"\"#0000FF\">");
        out.println("<p align="+"\"center\"><font size="+"\"5\"
color="+"\"#0000FF\"><b><i>Distributed Computer Aided Prototyping
System</i></b></font></p>");
        out.println("<p align="+"\"center\"><font size="+"\"5\"
color="+"\"#0000FF\"><b><i>Upload Project Files</i></b></font></p>");

        out.println("<hr color="+"\"#0000FF\"><br>");
        out.println("<body background=\"/CapsWeb/bg2.gif\">");

}


private void bottomOfPage(){

        out.println("<br><br><br>"+"<hr color="+"\"#0000FF\">");


        out.println("<center><a
href="+"\"/CapsWeb/servlet/logout\">Logout</a></center>");



        out.println("</html>");




}
}
```

137

THIS PAGE INTENTIONALLY LEFT BLANK

# APPENDIX B CAPS EDITOR CHANGES

```java
package caps.AdaFileEditor;

import javax.swing.*;

/**
  * The MenuBar of the Graph Editor.
  *
  * @author Shen-Yi Tao
  * @version 1.1
  */
public class TextEditorMenuBar extends JMenuBar
{
   //add  1/2/99 SYT
    private FileMenu aFileMenu;


   /**
     * The constructor for this class.
     */
   public TextEditorMenuBar (TextEditor parent)
   {
      super ();
      //add SYT 1/2/00
      aFileMenu = new FileMenu (parent);

      add (aFileMenu);

    }


} // End of the class EditorMenuBar
```

```
/*
 * TextEditor.java
 *
 * Created on August 16, 2001, 10:40 AM
 */

package caps.AdaFileEditor;


import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;
import java.io.*;
import java.io.File;


/**
 *
 * @author   administrator
 * @version
 */
public class TextEditor extends JFrame implements Runnable, ActionListener, KeyListener{

    protected JEditorPane editor;

    private File file;

    protected TextEditorMenuBar aTextEditorMenuBar;

    /**
    * The initial width of the GraphEditor
    */
    private final int INITIAL_WIDTH = 500;

   /**
    * The initial height of the Graph Editor
    */
    private final int INITIAL_HEIGHT = 600;

    boolean dirty;
    boolean neverNamed;




    /** Creates new TextEditor */
    public TextEditor() {

       dirty = false;
       neverNamed = true;
    }

    public void run() {

        initialize();

    }

    public void initialize(){

        file = new File("Untitled");

        try{
            UIManager.setLookAndFeel (UIManager
            .getCrossPlatformLookAndFeelClassName ());
        }
        catch (Exception e)
        {
```

140

```java
            System.err.println("Error loading L&F: " + e);
        }

        setTitle ("Web-CAPS Text Editor:  " + file.getName() );
        setDefaultCloseOperation (WindowConstants.DO_NOTHING_ON_CLOSE);
        addWindowListener( new ExitTextEditor(this) );


        setSize (INITIAL_WIDTH, INITIAL_HEIGHT);
        Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
        setLocation((screenSize.width - INITIAL_WIDTH) / 2,
        (screenSize.height - INITIAL_HEIGHT) / 2);

        aTextEditorMenuBar = new TextEditorMenuBar(this);

        setJMenuBar(aTextEditorMenuBar);


        editor = this.createEditorPane();
        editor.addKeyListener(this);

        JScrollPane p1 = new JScrollPane (editor,
                    ScrollPaneConstants.VERTICAL_SCROLLBAR_AS_NEEDED,
                    ScrollPaneConstants.HORIZONTAL_SCROLLBAR_AS_NEEDED );

        getContentPane ().add (p1);

        setVisible (true);


    }


    private JEditorPane createEditorPane(){

        JEditorPane editorPane = new JEditorPane();
        editorPane.setEditable(true);
        editorPane.setContentType("text/plain");
        String s = null;

        return editorPane;
    }

    public void openFile(){

        if(dirty){

            int selected = JOptionPane.showConfirmDialog ( null
            , "Choose \"Yes\" to continue without saving.\n"
            + "Choose \"No\" to cancel.", "Warning"
            , JOptionPane.YES_NO_OPTION
            , JOptionPane.WARNING_MESSAGE);

            if (selected == JOptionPane.YES_OPTION){

                 String part;
                 StringBuffer all = new StringBuffer("");

                JFileChooser fileChooser = new JFileChooser();
                fileChooser.setFileSelectionMode(JFileChooser.FILES_ONLY);
                int result = fileChooser.showOpenDialog(this);
                if( result == JFileChooser.CANCEL_OPTION ){

                }
                if(result == JFileChooser.APPROVE_OPTION ){
                    file = fileChooser.getSelectedFile();

                    if(file != null){
```

```
            try{

                FileReader fr = new FileReader(file);
                BufferedReader br= new BufferedReader(fr);
                while( (part=br.readLine()) != null){

                    all.append(part);
                    all.append('\n');
                }

                br.close();
                fr.close();

                editor.setText(all.toString());
                neverNamed = false;
                dirty = false;
            }
            catch( IOException fnf ){

            }

            setTitle("Web-CAPS Text Editor:  "+ file.getName());
            repaint();
        }

        }
    }

}
else{

    String part;
        StringBuffer all = new StringBuffer("");

        JFileChooser fileChooser = new JFileChooser();
        fileChooser.setFileSelectionMode(JFileChooser.FILES_ONLY);
        int result = fileChooser.showOpenDialog(this);
        if( result == JFileChooser.CANCEL_OPTION ){

        }
        if(result == JFileChooser.APPROVE_OPTION ){
            file = fileChooser.getSelectedFile();
            if(file != null){

            try{

                FileReader fr = new FileReader(file);
                BufferedReader br= new BufferedReader(fr);
                while( (part=br.readLine()) != null){

                    all.append(part);
                    all.append('\n');
                }

                br.close();
                fr.close();

                editor.setText(all.toString());
                neverNamed = false;
                dirty = false;
            }
            catch( IOException fnf ){

            }

            setTitle("Web-CAPS Text Editor:  "+ file.getName());
            repaint();
        }
        }
```

```java
        }



    }//end of OpenFile

    public void saveFile(){

        if(neverNamed){

            JFileChooser fileChooser = new JFileChooser();
            fileChooser.setFileSelectionMode(JFileChooser.FILES_ONLY);
            int result = fileChooser.showSaveDialog(this);
            if( result == JFileChooser.CANCEL_OPTION ){


            }
            if(result == JFileChooser.APPROVE_OPTION){
            file = fileChooser.getSelectedFile();

                if( file != null ){

                String all = editor.getText();
                try{
                    FileWriter fw = new FileWriter(file);
                    BufferedWriter bw = new BufferedWriter(fw);
                    bw.write(all, 0, all.length() );
                    bw.close();
                    fw.close();
                    dirty = false;

                }
                catch(IOException e){

                    System.out.println(e);

                }
                setTitle("Web-CAPS Text Editor:  "+ file.getName());
                repaint();

            }
            }


        }
        else{

            String all = editor.getText();
                try{
                    FileWriter fw = new FileWriter(file);
                    BufferedWriter bw = new BufferedWriter(fw);
                    bw.write(all, 0, all.length() );
                    bw.close();
                    fw.close();
                    dirty = false;

                }
                catch(IOException e){

                    System.out.println(e);

                }
             setTitle("Web-CAPS Text Editor:  "+ file.getName());
             repaint();
            }
```

```java
}//end of saveFile

 public void saveAs(){
    //System.out.println("inside saveAS");
    JFileChooser fileChooser = new JFileChooser();
    fileChooser.setFileSelectionMode(JFileChooser.FILES_ONLY);

    fileChooser.setSelectedFile(file);
    int result = fileChooser.showSaveDialog(this);
    if( result == JFileChooser.CANCEL_OPTION ){


    }
    if(result == JFileChooser.APPROVE_OPTION){

        file = fileChooser.getSelectedFile();


        if( file != null ){

        neverNamed = false;
        saveFile();

    }
    }


}

public void newFile(){

    if(safeToClose()){

        dirty = false;
        neverNamed = true;
        file = new File("Untitled");
        setTitle("Web-CAPS Text Editor:  "+ file.getName());
        editor.setText("");
        repaint();

    }


}

public void closeEditor(){

    if(safeToClose()){

        dispose();

    }
    else{

        int selected = JOptionPane.showConfirmDialog ( null
          , "Changes made to file have not been saved.\n"
          + "Choose \"Yes\" to continue without saving.\n"
          + "Choose \"No\" to cancel.", "Warning"
          , JOptionPane.YES_NO_OPTION
          , JOptionPane.WARNING_MESSAGE);

        if (selected == JOptionPane.YES_OPTION){

            dispose();

        }
```

144

```java
            }


        }
    public boolean safeToClose(){

        if(dirty){

             return false;

        }
        else{

            return true;

        }


    }


     public void actionPerformed(java.awt.event.ActionEvent p1) {
     }
     public void keyReleased(KeyEvent p1) {

     }

     public void keyPressed(KeyEvent p1) {

     }


     public void keyTyped(KeyEvent p1) {

         dirty = true;
        // if(neverNamed){

             //setTitle("Web-CAPS Text Editor");

         //}
        // else{

             setTitle("Web-CAPS Text Editor:  "+ file.getName()+"*");

          //}

     }

}
```

```java
/*
 * QueryString.java
 *
 * Created on August 6, 2001, 12:58 PM
 */

package caps.FileManager;

import java.net.*;
/**
 *
 * @author  This class was taken from "JAVA Network Programming by Elliotte Harold
 * @version
 */
public class QueryString extends Object {

    private String query;

    /** Creates new QueryString */
    public QueryString( Object name, Object value ) {

        query = URLEncoder.encode(name.toString()) + "=" +
URLEncoder.encode(value.toString());

    }

    public QueryString(){

        query = "";

    }

    public synchronized void add(Object name, Object value){

        if(!query.trim().equals("")) query += "&";
        query += URLEncoder.encode(name.toString()) + "=" +
URLEncoder.encode(value.toString());

    }

    public String toString(){

        return query;

    }

}
```

```java
package caps.CAPSMain;

import javax.swing.*;
import javax.swing.filechooser.FileSystemView;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.File;
import java.util.Vector;
import java.io.*;
import java.net.*;
import java.lang.*;
import java.util.*;
import java.util.Date;

/**
  * This class holds the 'Prototype' menu items.
  *
  * @author Shen-Yi Tao
  * @version 1.1
  */

/**
  * Changes:
  *     7-9-99 MTS
  *     add statement to set protoHome, protoName and protoVersion
  *     of CAPSMainWindow object in processNewMenuItem and
  *     processOpenMenuItem
  *
  *     7-12-99 MTS
  *     change code in processNewMenuItem and processOpenMenuItem
  *     so that user only have to specify the $HOME directory
  *     instead of $HOME/.caps using the -DPROTOTYPEHOME flag
  *
  */

public class PrototypeMenu extends JMenu implements ActionListener
{

    static final int BUFF_SIZE = 100000;
    static final byte[] buffer = new byte[BUFF_SIZE];
    /**
      * Initiates the 'New' event
      */
    //private JMenuItem newMenuItem = new JMenuItem ("New");

    /**
      * Initiates the 'Open' event
      */
    //private JMenuItem openMenuItem = new JMenuItem ("Open Local Copy");


    /**
      * Initiates the 'Commit Work' event
      */
    //private JMenuItem commitWorkMenuItem = new JMenuItem ("Commit Work to Server");

    /**
      * Initiates the 'Download from Application Server' event
      */
    //private JMenuItem downloadMenuItem = new JMenuItem ("Download Project from Server");

    /**
      * Initiates the 'File Manager' event
      */
    private JMenuItem initiateFileManager = new JMenuItem ("File Manager");


    /**
      * Initiates the 'Retrieve From DDB' event
```

147

```java
   */
//private JMenuItem retrieveMenuItem = new JMenuItem ("Retrieve From DDB");

/**
  * Initiates the 'Quit' event
  */
private JMenuItem quitMenuItem = new JMenuItem ("Quit");

/**
  * The main window which owns this menu.
  */
protected CAPSMainWindow ownerWindow;

/**
  * Constructor for this class.
  *
  * @param owner The main window which has created this menu.
  */
public PrototypeMenu (CAPSMainWindow owner)
{
    super ("Prototype");

    ownerWindow = owner;
    loadPrototype();

    //add (newMenuItem);
    //add (openMenuItem);
   // add (commitWorkMenuItem);
    //add (downloadMenuItem);
    add(initiateFileManager);

    add (quitMenuItem);

    initiateFileManager.setToolTipText("Open File Manager");
    quitMenuItem.setToolTipText("Exit Web-Caps");

    /*
     * Register the action listeners
     */
    //newMenuItem.addActionListener (this);
    //openMenuItem.addActionListener (this);
    //commitWorkMenuItem.addActionListener (this);
    //downloadMenuItem.addActionListener (this);
    initiateFileManager.addActionListener(this);
    quitMenuItem.addActionListener (this);
}


/**
 *
 *
 *
 *
 */
public void loadPrototype(){

    //System.out.println("Entering loadPrototype");

    boolean projectExists = false;
    //System.out.println(projectExists);

    String protoHome = System.getProperty ("PROTOTYPEHOME");
    File protoDir;
    String proto = ownerWindow.getProtoName();
    String version = ownerWindow.getProtoVersion();
    String name = proto;

    if (protoHome == null){       // If it is not set as a command line argument
```

```
        //System.out.println("Entering 1st if");
    File homeDir = FileSystemView.getFileSystemView ().getHomeDirectory ();

    protoHome = homeDir.toString();
    protoDir = new File (protoHome + File.separator + ".caps");
    if (!protoDir.exists ())
        protoDir.mkdir ();
}
else
{
        //System.out.println("Entering 1st else");
    // 7-12-99 add the arguments File.separator + ".caps" to
    // the following statement since protoHome now only
    // contains $HOME instead of $HOME/.caps.
    protoDir = new File (protoHome + File.separator + ".caps");
    //protoDir = new File (protoHome);
    if (!protoDir.exists ())
        protoDir.mkdir ();
}

Vector prototypeNames = new Vector (0, 2);
File [] dirs = protoDir.listFiles ();
String protoName = "";

if (dirs.length == 0){
        //System.out.println("Entering if:  dirs.length = 0");
    File directory = new File (protoHome + File.separator + ".caps"
                            + File.separator + proto + File.separator
                            + version + File.separator );
    directory.mkdirs();
    File file = new File (protoHome + File.separator + ".caps"
                            + File.separator + proto + File.separator
                            + version + File.separator + name + ".psdl");
    //add 8/26/00 SYT
    File adaTemplet = new File (protoHome + File.separator  + ".caps"
                            + File.separator + proto + File.separator
                            + version + File.separator  );

    try
        {
            file.delete ();
            file.createNewFile ();
            adaTemplet.delete();
            adaTemplet.createNewFile();
        }
        catch (java.io.IOException ex)
        {
            System.out.println (ex);
        }
        ownerWindow.setPrototype (file);
        ownerWindow.setAdaTemplet(adaTemplet);
        ownerWindow.setProtoHome(protoHome);
        ownerWindow.setProtoName(proto);
        ownerWindow.setProtoVersion(version);
        return;
  }
for (int ix = 0; ix < dirs.length; ix++){

    protoName = dirs [ix].getName ();
    File subDirs [] = dirs [ix].listFiles ();
    for (int jx = 0; jx < subDirs.length; jx++)
    {
        prototypeNames.addElement (protoName.concat (File.separator
                                        + subDirs [jx].getName ()));
    }

}
    String protoVersion = proto + File.separator + version;
```

149

```java
      for (Enumeration f = prototypeNames.elements() ; f.hasMoreElements() ;) {
              String tempProto = "";
              tempProto = (String)f.nextElement();
              if(tempProto.equals(protoVersion)){

                  projectExists = true;//the project exists and there is no need to
create a new .psdl file and adaTemplate

                  break;

              }

      }
      //System.out.println(projectExists);

       if( projectExists == false ){
           //System.out.println("Entering if: projectExists = false");

           //create new prototype folders and files
           File directory1 = new File (protoHome + File.separator + ".caps"
                               + File.separator + proto + File.separator
                               + version  );
           directory1.mkdirs();
           File file = new File (protoHome + File.separator + ".caps"
                               + File.separator + proto + File.separator
                               + version + File.separator + name + ".psdl");
           String temporaryFile = "";
           temporaryFile = file.toString();
           //System.out.println(temporaryFile);
           File adaTemplet = new File (protoHome + File.separator  + ".caps"
                               + File.separator + proto + File.separator
                               + version + File.separator  );

           try
               {
                  file.delete ();
                  file.createNewFile ();
                  adaTemplet.delete();
                  adaTemplet.createNewFile();
               }
               catch (java.io.IOException ex)
               {
                  System.out.println (ex);
               }

               ownerWindow.setPrototype (file);
               ownerWindow.setAdaTemplet(adaTemplet);
               ownerWindow.setProtoHome(protoHome);
               ownerWindow.setProtoName(proto);
               ownerWindow.setProtoVersion(version);
          }
        else{
             //System.out.println("Entering last else");
             File selectedDir = new File (protoHome + File.separator + ".caps"
                                     + File.separator + protoVersion);

             File file = new File (selectedDir.getAbsolutePath ()
                                   + File.separator + selectedDir
                                      .getParentFile ().getName () + ".psdl");

             File adaTemplet = new File (selectedDir.getAbsolutePath () );


              if (!file.exists ())
                  System.out.println("Prototype could not be opened.");

              ownerWindow.setPrototype (file);
              //add 8/26/00 SYT
              ownerWindow.setAdaTemplet(adaTemplet);
```

```
                    // 7-9-99 MTS
                    // add statement to invoke setProtoHome, setProtoName,
                    // and setProtoVersion
                    ownerWindow.setProtoHome(protoHome);
                    ownerWindow.setProtoName(selectedDir.getParentFile().getName());
                    ownerWindow.setProtoVersion(selectedDir.getName());

        }


          //System.out.println("Leaving loadPrototype");

    }

    /**
      * Action event handler for the menu events.
      *
      * @param e The action event that is created by selecting
      *  a menu item from this menu
      */
    public void actionPerformed(ActionEvent e)
    {
        //if (e.getSource () == newMenuItem)
        //{
            //processNewMenuItem ();
      //  }
        //else if (e.getSource () == openMenuItem)
        //{
          // processOpenMenuItem ();
      //  }
        //else if (e.getSource () == downloadMenuItem)
        //{
            //try{


//downLoadPrototype("http://131.120.9.87:8080/CapsAdmin/servlet/testDownload",
"c:\\TEMP\\jamcdona\\temp");
                //System.out.println("Not working at this time.");
            //}
            //catch(Exception f){
                //System.out.println(f);
            //}

            //downLoadPrototype(String from, String to);
      //}
        if (e.getSource() == initiateFileManager){

            ownerWindow.manageFiles();

        }
        else if (e.getSource () == quitMenuItem)
        {
            // Exit the program if all of the prototypes are saved.
            if (ownerWindow.isOpenPrototypeSaved ())
                System.exit (0);
        }
    }

    /**
      * Handles the event which is caused by selecting the 'New' menu item.
      */
    public void processNewMenuItem ()
    {
        // The system property for the home prototype directory.
        String protoHome = System.getProperty ("PROTOTYPEHOME");
        File protoDir;
        // 7-9-99 MTS
        // add local variable proto and version
        String proto;
```

```
         String version;

         if (protoHome == null)
         {      // If it is not set as a command line argument
            File homeDir = FileSystemView.getFileSystemView ().getHomeDirectory ();
            //protoHome = new String (homeDir + File.separator + ".caps");
            //protoDir = new File (protoHome);
            protoHome = homeDir.toString();
            protoDir = new File (protoHome + File.separator + ".caps");
            if (!protoDir.exists ())
               protoDir.mkdir ();
         }
         else
         {
            // 7-12-99 add the arguments File.separator + ".caps" to
            // the following statement since protoHome now only
            // contains $HOME instead of $HOME/.caps.
            protoDir = new File (protoHome + File.separator + ".caps");
            //protoDir = new File (protoHome);
            if (!protoDir.exists ())
               protoDir.mkdir ();
         }

         // 7-9-99 MTS
         // moved String proto declaration to beginning of method
         // String proto = JOptionPane.showInputDialog (ownerWindow,

         proto = JOptionPane.showInputDialog (ownerWindow,
                  "Enter Prototype Name : ", "New", JOptionPane.PLAIN_MESSAGE);
         if (proto == null)
            return;

         // 7-9-99 MTS
         // moved String version declaration to beginning of method
         //String version = JOptionPane.showInputDialog (ownerWindow,

         version = JOptionPane.showInputDialog (ownerWindow,
                  "Prototype Version Information : ","New"
                  , JOptionPane.PLAIN_MESSAGE);
         {
            String name = proto;

            // 7-12-99 add the arguments File.separator + ".caps" to
            // the following statement since protoHome now only
            // contains $HOME instead of $HOME/.caps.
            File file = new File (protoHome + File.separator + ".caps"
                               + File.separator + proto + File.separator
                               + version + File.separator + name + ".psdl");
            //add 8/26/00 SYT
            File adaTemplet = new File (protoHome + File.separator  + ".caps"
                               + File.separator + proto + File.separator
                               + version + File.separator  );

            if (file.exists ())
            {
               int selected = JOptionPane.showConfirmDialog (ownerWindow
                           , "Selected prototype file already exists.\n"
                           + "Do you want to overwrite it ?");
               if (selected == JOptionPane.YES_OPTION)
               {
                  try
                  {
                     file.delete ();
                     file.createNewFile ();
                     adaTemplet.delete();
                     adaTemplet.createNewFile();
                  }
                  catch (java.io.IOException ex)
                  {
```

152

```java
                System.out.println (ex);
            }
            ownerWindow.setPrototype (file);
            //add 8/26/00 SYT
            ownerWindow.setAdaTemplet(adaTemplet);

            // 7-9-99 MTS
            // add statement to invoke setProtoHome, setProtoName,
            // and setProtoVersion
            ownerWindow.setProtoHome(protoHome);
            ownerWindow.setProtoName(proto);
            ownerWindow.setProtoVersion(version);
        }
    }
    else {
        try {
            File dir = file.getParentFile ().getParentFile ();
            dir.mkdir ();
            File vers = file.getParentFile ();
            vers.mkdir ();
            file.createNewFile ();
            adaTemplet.createNewFile();

        }
        catch (java.io.IOException ex)
        {
            System.out.println (ex);
        }
        ownerWindow.setPrototype (file);
        //add 8/26/00 SYT
        ownerWindow.setAdaTemplet(adaTemplet);
        // 7-9-99 MTS
        // add statement to invoke setProtoHome, setProtoName,
        // and setProtoVersion
        ownerWindow.setProtoHome(protoHome);
        ownerWindow.setProtoName(proto);
        ownerWindow.setProtoVersion(version);
    }
  }
}

/**
  * Handles the event which is caused by selecting the 'Open' menu item.
  */
public void processOpenMenuItem ()
{

    String protoHome = System.getProperty ("PROTOTYPEHOME");
    File protoDir;
    if (protoHome == null)
    {   // If it is not set as a command line argument
       File homeDir = FileSystemView.getFileSystemView ().getHomeDirectory ();
       //protoHome = new String (homeDir + File.separator + ".caps");
       //protoDir = new File (protoHome);
       protoHome = homeDir.toString();
       protoDir = new File (protoHome +  File.separator + ".caps" );
       if (!protoDir.exists ())
          protoDir.mkdir ();
    }
    else
    {
       // 7-12-99 add the arguments File.separator + ".caps" to
       // the following statement since protoHome now only
       // contains $HOME instead of $HOME/.caps.
       protoDir = new File (protoHome + File.separator + ".caps");
       if (!protoDir.exists ())
          protoDir.mkdir ();
    }
```

153

```
        Vector prototypeNames = new Vector (0, 2);
        File [] dirs = protoDir.listFiles ();
        String protoName = "";

        if (dirs.length == 0)
        {
            JOptionPane.showMessageDialog
               (ownerWindow, "No prototype is found to open",
                                    "Error Message", JOptionPane.ERROR_MESSAGE);
        }
        else
        {
            for (int ix = 0; ix < dirs.length; ix++)
            {
                protoName = dirs [ix].getName ();
                File subDirs [] = dirs [ix].listFiles ();
                for (int jx = 0; jx < subDirs.length; jx++)
                {
                    prototypeNames.addElement (protoName.concat (File.separator
                                                  + subDirs [jx].getName ()));
                }

            }

            Object [] protos = prototypeNames.toArray ();
            String selected = (String) JOptionPane
                            .showInputDialog (ownerWindow
                              , "Select a protoype : ", "Open"
                              , JOptionPane.INFORMATION_MESSAGE, null
                              , protos, protos [0]);

            if (selected != null)
            {
                // 7-12-99 add the arguments File.separator + ".caps" to
                // the following statement since protoHome now only
                // contains $HOME instead of $HOME/.caps.
                File selectedDir = new File (protoHome + File.separator + ".caps"
                                         + File.separator + selected);
                File file = new File (selectedDir.getAbsolutePath ()
                                     + File.separator + selectedDir
                                        .getParentFile ().getName () + ".psdl");
                //add 8/26/00 SYT
              /*File adaPath = new File (protoHome + protoHome + File.separator
                                  + ".caps" + File.separator + proto
                                  + File.separator
                                  + version + File.separator);
                */
                File adaTemplet = new File (selectedDir.getAbsolutePath () );


// 4/30/99 MTS
// debug statements
//System.out.println ("Prototype Home = " + protoHome);
//System.out.println ("Directory name = " + selectedDir.toString());
//System.out.println ("Prototype name = " + file.toString());

                if (!file.exists ())
                    JOptionPane.showMessageDialog (ownerWindow
                                  , "The selected prototype file cannot be opened"
                                  , "Error Message"
                                  , JOptionPane.ERROR_MESSAGE);
                ownerWindow.setPrototype (file);
                //add 8/26/00 SYT
                ownerWindow.setAdaTemplet(adaTemplet);
                // 7-9-99 MTS
                // add statement to invoke setProtoHome, setProtoName,
                // and setProtoVersion
                ownerWindow.setProtoHome(protoHome);
                ownerWindow.setProtoName(selectedDir.getParentFile().getName());
```

154

```
                    ownerWindow.setProtoVersion(selectedDir.getName());
                }
            }
        }






    }
```

```java
/*
 * Poster.java
 *
 * Created on August 6, 2001, 1:07 PM
 */

package caps.FileManager;

import java.net.*;
import java.io.*;

/**
 *
 * @author  This class was taken from "JAVA Network Programming by Elliotte Harold
 * @version
 */
public class Poster extends Object {

    private URL url;

    private QueryString query = new QueryString();

    /** Creates new Poster */
    public Poster( URL url ) {

        this.url = url;

    }

    public void add( String name, String value ){

        query.add(name,value);

    }

    public URL getURL(){

        return this.url;

    }

    public InputStream post() throws IOException{

        URLConnection uc = url.openConnection();
        uc.setDoOutput(true);
        OutputStreamWriter out = new OutputStreamWriter(uc.getOutputStream(), "ASCII");

        out.write(query.toString());
        //out.write("r\\n\\");
        out.flush();
        out.close();

        return uc.getInputStream();

    }


}
```

156

```java
/*
 * Manager.java
 *
 * Created on August 4, 2001, 10:09 PM
 */

package caps.FileManager;


import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import javax.swing.event.*;
import javax.swing.ScrollPaneConstants.*;
import java.io.*;
import java.util.Enumeration;
import java.util.*;
import java.net.*;
import java.util.Date;
import java.util.Arrays;
import java.io.File;

/**
 *
 * @author  JAM
 * @version
 */
public class Manager extends JFrame implements Runnable, ActionListener {

    static final int BUFF_SIZE = 100000;
    static final byte[] buffer = new byte[BUFF_SIZE];
    /**
     * The panel that holding the tables
     */
    protected JPanel panel;

    /**
     * The panel that holds buttons
     */
    protected JPanel buttonPanel;

    /**
     * Includes the local and server tables.
     */
    protected JSplitPane innerSplit;

     /**
      * The table that displays the local project files
      */
     protected JTable localTable;

    /**
     * The label for local project files
     */
     protected JLabel localLabel;

    /**
     * The table that displays the server project files
     */
     protected JTable serverTable;

    /**
     * The label for server project files
     */
     protected JLabel serverLabel;

    /**
     * Button to use local files
     */
```

157

```
 protected JButton localButton;

/**
 * Button to use download files
 */
 protected JButton downLoadButton;

/**
 * Button to use upload files
 */
 protected JButton upLoadButton;

/**
 * Button to use upload files
 */
 protected JButton closeButton;
/**
 * The initial width of the GraphEditor
 */
 private final int INITIAL_WIDTH = 800;

/**
 * The initial height of the Graph Editor
 */
 private final int INITIAL_HEIGHT = 600;

 private File[] localFiles=null;
 private File[] serverFiles=null;

 private File folderOfLocalFiles;
 private String appserverUrl;//codebase for server to allow for communication
 private String capsuserFolder;//where user folders are kept on server
 private String servletFolder;//where download servlet is located on server relative
to codebase
 private String usersFolder;
 private String user;
 private String name;
 private String ver;

 /** Creates new Manager */
 public Manager( File adaTemplate, String url, String u, String n, String v ) {

     folderOfLocalFiles = adaTemplate;
     appserverUrl = url;
     capsuserFolder = "/capsusers";
     servletFolder = "/servlet/testDownload";
     user = u;
     name = n;
     ver = v;
     usersFolder = "/"+user+"/"+name+"/"+ver;

 }

 // this is another thread to paint main window of File Manager
 public void run ()
 {
     initialize ();
 }

/**
 * The initialization of the GUI takes place here
 *
 * 5 Aug 01 – JAM
 */
 public void initialize ()
 {

     // Set the look and feel to a platform independent view
     try
```

158

```
            {
                UIManager.setLookAndFeel (UIManager
                .getCrossPlatformLookAndFeelClassName ());
            } catch (Exception e)
            {
                System.err.println("Error loading L&F: " + e);
            }

        setTitle ("Web-CAPS File Manager:  User:  "+user+"  Project:  "+name+"  Version:
"+ver );
        setDefaultCloseOperation (WindowConstants.DO_NOTHING_ON_CLOSE);
        addWindowListener( new ExitManager(this) );


        setSize (INITIAL_WIDTH, INITIAL_HEIGHT);
        Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();
        setLocation((screenSize.width - INITIAL_WIDTH) / 2,
        (screenSize.height - INITIAL_HEIGHT) / 2);

        BorderLayout bLayout = new BorderLayout ();

        bLayout.setVgap (3);

        //retrieves file info from server
        serverFiles = checkFilesServer( appserverUrl + servletFolder, usersFolder,
serverFiles );

        localFiles = getLocalFiles(folderOfLocalFiles);

        //retrieve info to fill server JTable
        Object[][] sFiles = createFileArray(serverFiles);
        Object[][] lFiles = createFileArray(localFiles);

        String[] localNames = {"Local Files", "Last Modified"};
        String[] serverNames = {"Server Files", "Last Modified"};

        localTable = new JTable( lFiles, localNames );
        //localTable.setSelectionMode(ListSelectionModel.MULTIPLE_INTERVAL_SELECTION);
        //localTable.setCellSelectionEnabled(true);
        serverTable = new JTable( sFiles, serverNames );
        serverTable.setSelectionMode(ListSelectionModel.MULTIPLE_INTERVAL_SELECTION);
        serverTable.setCellSelectionEnabled(true);

        panel = new JPanel (bLayout, true);

        panel.setAlignmentX (LEFT_ALIGNMENT);
        panel.setAlignmentY (TOP_ALIGNMENT);
        panel.setBorder (BorderFactory.createLoweredBevelBorder ());

        getContentPane ().add (panel);

        localButton = new JButton("Use Local Files");
        localButton.setBorder(BorderFactory.createRaisedBevelBorder());
        downLoadButton = new JButton("DownLoad Files from Server");
        downLoadButton.setBorder(BorderFactory.createRaisedBevelBorder());
        //upLoadButton = new JButton("Upload Files to Server");
        //upLoadButton.setBorder(BorderFactory.createRaisedBevelBorder());
        closeButton = new JButton("Close File Manager");
        closeButton.setBorder(BorderFactory.createRaisedBevelBorder());

        localButton.addActionListener(this);
        downLoadButton.addActionListener(this);
        //upLoadButton.addActionListener(this);
        closeButton.addActionListener(this);

        buttonPanel = new JPanel();
        buttonPanel.setBorder (BorderFactory.createLoweredBevelBorder ());
        buttonPanel.add(localButton);
        buttonPanel.add(downLoadButton);
```

```
            //buttonPanel.add(upLoadButton);
            buttonPanel.add(closeButton);

            JScrollPane p1 = new JScrollPane (localTable,
    ScrollPaneConstants.VERTICAL_SCROLLBAR_AS_NEEDED,
    ScrollPaneConstants.HORIZONTAL_SCROLLBAR_AS_NEEDED );
            // p1.setBackground (Color.white);
            JScrollPane p2 = new JScrollPane (serverTable,
    ScrollPaneConstants.VERTICAL_SCROLLBAR_AS_NEEDED,
    ScrollPaneConstants.HORIZONTAL_SCROLLBAR_AS_NEEDED);
            //p2.setBackground (Color.white);
            //p2.getViewport().setBackground(Color.white);//added 02/07/01 to fix display
    problem--jam

            innerSplit = new JSplitPane (JSplitPane.HORIZONTAL_SPLIT, p1, p2);
            innerSplit.setContinuousLayout(true);
            innerSplit.setOneTouchExpandable (true);
            innerSplit.setDividerLocation (getWidth () / 2);
            innerSplit.setBorder (BorderFactory.createLoweredBevelBorder ());

            panel.add (innerSplit, BorderLayout.CENTER);
            panel.add (buttonPanel, BorderLayout.NORTH);


            setVisible (true);


        }//end of initialize


    /**
     * This method is used to copy files from the server to the local system.
     *
     * 13 Aug 01 - JAM
     */
        public static void copyDown(String from, File toDir, File[] server )throws
    IOException{
            String fileName = "";

            String to = toDir.toString();

            //System.out.println(to);

            for( int iw = 0; iw < server.length; iw++){

                //System.out.println ("Inside copy down for loop");

                java.net.URL url = new URL( from+server[iw].getName() );

                //System.out.println ("url found");

                InputStream in = url.openStream();

                OutputStream out = null;
                try{

                    File temp = new File( to + File.separator + server[iw].getName() );

                    temp.createNewFile();

                    //System.out.println( "file created" );

                    out = new FileOutputStream( to + File.separator + server[iw].getName());

                    while(true){
                        synchronized(buffer){
                            int amountRead = in.read(buffer);
                            if(amountRead == -1){
                                break;
```

160

```
                        }
                        out.write(buffer, 0, amountRead);
                    }
                }
            }finally{
                if(in!=null){
                    in.close();
                }
                if(out!=null){
                    out.close();
                }
            }
            System.out.println( "File:  "+server[iw].getName()+" saved locally." );
        }//end of for loop


    } // End of the class PrototypeMenu

    /**
     *This function queries the servlet on the server and copies a file to the users
folder
     *
     *
     */
    public static void upLoadFile(String from){

        URL url = null;
        String userFolder = "mcdonald\\elevator_control_system\\version_1\\";
        String fileToSave = "d:\\caps\\elevator_control_system.elevator_1_23.at";
        String first = "James";
        try{

            url = new URL(from);
            System.out.println ("Inside first try");
        }
        catch(MalformedURLException g){
            System.out.println("Inside first catch");
            System.out.println (g);

        }
        Poster postFile = new Poster(url);
        postFile.add("first", first);
        postFile.add("File", fileToSave);

        try{

            InputStream in = postFile.post(); //sends data to servlet and returns the
InputStream
            System.out.println ("Inside second try");
           // InputStreamReader p = new InputStreamReader(in, "ASCII");


            //while(true){

                //int data = p.read();
                //if(data == -1){

                        //break;

                //}
                System.out.println(in);

       // }
            //p.close();
            in.close();

            //System.out.println ("Inside InputStream");
        }
        catch(IOException e){
```

```java
            System.err.println(e);

        }



    }//end of upLoadFile


    /**
     * This function queries the servlet on the server and returns a File[] of
     * the project folder
     *
     * 13 Aug 01 - JAM
     */
    public static File[] checkFilesServer(String from, String folder, File[] server){

        URL url=null;

        String userFolder = folder;//"mcdonald\\elevator_control_system\\version_1\\";

        try{

            url = new URL(from);

        }
        catch(MalformedURLException g){

        }

        Poster postIt = new Poster(url);

        postIt.add("userFolder", userFolder);//used to send parameters to servlet

        //System.out.println ("Inside copy");


        try{

            InputStream in = postIt.post(); //sends data to servlet and returns the
InputStream

            //System.out.println ("Inside InputStream");

            ObjectInputStream p = new ObjectInputStream(in);//ObjectInputStream to read
File[] returned from servlet

            try{

                server = (File[])p.readObject();

                //File protoDir = new File("\\Temp");

                String protoName = "";

                long time = 0;

                //takes returned File[] and spits out name and lastmodifed info
                for (int ix = 0; ix < server.length; ix++){

                    protoName = server[ix].getName ();
                    time      = server[ix].lastModified();
                    java.util.Date t = new Date(time);
                    String tt = t.toString();
                    //System.out.println("File Name:  "+protoName+"  Last Modified:
"+tt);
```

162

```java
                }

            }
            catch(ClassNotFoundException e){

                System.out.println(e);

            }

            in.close();//closes down to input stream

        }
        catch(IOException e){

            System.err.println(e);

        }

        return server;

    }//end of checkFilesServer

    public void actionPerformed(ActionEvent e){

        if (e.getSource () == localButton){

            System.out.println("Local Button Activated");

            JOptionPane.showMessageDialog(null, "Your Local Project Files will be
used.");

            dispose();
        }
        else if (e.getSource() == downLoadButton){

            //System.out.println("Download Button Activated");

            int selected = JOptionPane.showConfirmDialog ( null
            , "Selecting this option will overwrite local files.\n"
            + "Do you want to overwrite it ?");
            if (selected == JOptionPane.YES_OPTION){

                //System.out.println("Inside Yes_Option");
                try{

                    copyDown(appserverUrl + capsuserFolder + usersFolder +"/",
folderOfLocalFiles, serverFiles);

                }
                catch(IOException ee){

                    System.out.println(ee);

                }
            }

        }
        /*
        else if (e.getSource() == upLoadButton){

            System.out.println("Upload Button Activated");

            int select = JOptionPane.showConfirmDialog ( null
                                    , "Selecting this option will overwrite server
files.\n"
                                    + "Do you want to overwrite it ?");

            if (select == JOptionPane.YES_OPTION){
```

163

```java
                    System.out.println("Inside Yes_Option");
                    upLoadFile("http://131.120.9.87:8080/CapsWeb/servlet/UploadTest");
                }

        }*/
        else if (e.getSource() == closeButton){

            //System.out.println("Close Button Activated");

            int select = JOptionPane.showConfirmDialog ( null
                                    , "Selecting this option will close the File
Manager.\n"
                                    + "Do you want to close it ?");

            if (select == JOptionPane.YES_OPTION){

                dispose();
            }

        }

    }//end of actionPerformed

    /**
     * This method takes returns a 2D array for the JTable used
     *
     * 13 Aug 01 - JAM
     */
    public Object[][] createFileArray( File[] inputFileArray ){

        int total = inputFileArray.length;
        //System.out.println(total);

        Object[][] files = new String[total][2];

        for(int iv = 0; iv < total; iv++){

            String name = inputFileArray[iv].getName();
            long time   = inputFileArray[iv].lastModified();
            java.util.Date t = new Date(time);
            String tt = t.toString();
            files[iv][0] = name;
            files[iv][1] = tt;

        }

        return files;

    }//end of createFileArray

    /**
     * This method returns a File[] for a local directory
     *
     * 13 Aug 01 - JAM
     */
    public File[] getLocalFiles( File localDir ){

        File local = new File( localDir.toString() );
        File[] list = local.listFiles();

        return list;


    }//end of getLocalFiles

}
```

```java
package caps.CAPSMain;

import javax.swing.*;
import java.awt.*;
import javax.swing.JMenu;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;

/**
 * This class implements the 'Help' menu.
 */
public class HelpMenu extends JMenu implements ActionListener
{
     private JMenuItem aboutMenuItem = new JMenuItem ("About Web-Caps");
     private JOptionPane about;
     private String msg = "Web-CAPS \n" +
                "Version 1.0 \n" +
                "Copyright (c) 2001 \n\n" +
                "Authors: \n" +
                "     Professor Man-tak Shing \n" +
                "     Major James A. McDonald USMC \n\n" +
                "Overview:  Web-CAPS is an internet based implementation \n" +
                "of the Naval Postgraduate School, Software Engineering\n" +
                "Group's Computer Aided Prototyping System:  CAPS.\n" +
                "CAPS is used to develop working prototypes of real-time \n"+
                "embedded systems.  \n"+
                "Go to http://wwwcaps.cs.nps.navy.mil for information about\n"+
                "using CAPS.\n";
     private ImageIcon aboutIcon;
   /**
     * Constructor for this class.
     */
    public HelpMenu ()
    {
       super ("Help");


       ClassLoader cl = this.getClass().getClassLoader();
       aboutIcon = new ImageIcon
                ( cl.getResource( "caps/Images/caps.gif" ) );


       about = new JOptionPane();
       add(aboutMenuItem);
       aboutMenuItem.addActionListener(this);
    }

   /**
     * Action event handler for the menu events.
     *
     * @param e The action event that is created by selecting
     * a menu item from this menu
     */
   public void actionPerformed(ActionEvent e)

   {
       if(e.getSource() == aboutMenuItem){

           about.showMessageDialog(null,msg,"About Web-
CAPS",JOptionPane.PLAIN_MESSAGE,aboutIcon);

       }

   }


} // End of the class HelpMenu
```

```java
/*
 * FileMenu.java
 *
 * Created on August 16, 2001, 11:17 AM
 */

package caps.AdaFileEditor;


import javax.swing.*;
import java.awt.event.*;
import java.awt.event.WindowEvent;
import java.io.*;
import java.io.File;


/**
 *
 * @author   administrator
 * @version
 */
public class FileMenu extends JMenu implements ActionListener {

    private File file;



    /**
      * Initiates the 'Save' event
      */
    private JMenuItem saveMenuItem = new JMenuItem ("Save");

    /**
      * Initiates the 'Save As' event
      */
    private JMenuItem saveAsMenuItem = new JMenuItem ("Save As...");

    /**
      * Initiates the 'Open' event
      */
    private JMenuItem openMenuItem = new JMenuItem ("Open");

    /**
      * Initiates the 'New' event
      */
    private JMenuItem newMenuItem = new JMenuItem ("New");

    /**
      * Initiates the 'Print' event
      */
    private JMenuItem printMenuItem = new JMenuItem ("Print");

    /**
      * Initiates the 'Exit' event
      */
    private JMenuItem exitMenuItem = new JMenuItem ("Exit");

    private TextEditor parent;

    /**
      * The constructor for the File menu
      */
    public FileMenu (TextEditor t) {
        super ("File");

        parent = t;
        add(newMenuItem);
        add(openMenuItem);
        add (saveMenuItem);
```

166

```java
      add(saveAsMenuItem);
      add (printMenuItem);
      add (exitMenuItem);

      newMenuItem.setToolTipText("Create New Text File");
      openMenuItem.setToolTipText("Open a Text File");
      saveMenuItem.setToolTipText ("Save the file");
      saveAsMenuItem.setToolTipText ("Save the file with a name of your choosing");
      printMenuItem.setToolTipText ("Print the Text File");
      exitMenuItem.setToolTipText ("Exit Text Editor");

      newMenuItem.addActionListener(this);
      openMenuItem.addActionListener(this);
      saveMenuItem.addActionListener (this);
      saveAsMenuItem.addActionListener (this);
      printMenuItem.addActionListener (this);
      exitMenuItem.addActionListener (this);
   }

   /**
     * Handles the menu events that occur when one of the menu items
     * is selected
     *
     * @param e The associated ActionEvent
     */
   public void actionPerformed (ActionEvent e)
   {
      /**
       * 11/3/99 SYT
       * This does not save file durning some condition.
       * Go to check out Editor.
       */
      if (e.getSource () == saveMenuItem)
      {
         parent.saveFile();
      }
      else if (e.getSource () == newMenuItem)
      {
         parent.newFile();
      }
      else if (e.getSource () == saveAsMenuItem)
      {

          parent.saveAs();
      }
      else if (e.getSource () == openMenuItem)
      {
         parent.openFile();
      }
      else if (e.getSource () == printMenuItem)
      {
         System.out.println ("Print has not yet been implemented");
      }
      else if (e.getSource () == exitMenuItem)
      {
            parent.closeEditor();

      }

   }
   /*
   private void openFile(){

      JFileChooser fileChooser = new JFileChooser();
      fileChooser.setFileSelectionMode(JFileChooser.FILES_ONLY);
      int result = fileChooser.showOpenDialog(this);
      if( result == JFileChooser.CANCEL_OPTION ){
          file = null;
      }
```

```
        else{
            file = fileChooser.getSelectedFile();
        }
    }
    */
} // End of the class FileMenu
```

```java
package caps.AdaFileEditor;

/*
 * Exit TextEditor.java
 *
 * Created on August 6, 2001, 10:47 AM
 */


import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;

/**
 *
 * @author  JAM
 * @version
 */
public class ExitTextEditor extends WindowAdapter {

    TextEditor text;
    /** Creates new ExitManager */
    public ExitTextEditor( TextEditor t ) {

        text = t;

    }

    /**
     * Window event handler file manager.
     *
     * @param e The window event that is created when the program
     *
     */
    public void windowClosing (WindowEvent e)
    {

        text.setVisible (false);
        text.dispose ();

      //System.exit (0);
    }

}
```

```java
/*
 * ExitManager.java
 *
 * Created on August 6, 2001, 10:47 AM
 */

package caps.FileManager;

import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;

/**
 *
 * @author  JAM
 * @version
 */
public class ExitManager extends WindowAdapter {

    Manager manager;
    /** Creates new ExitManager */
    public ExitManager( Manager m ) {

        manager = m;

    }

    /**
     * Window event handler file manager.
     *
     * @param e The window event that is created when the program
     *
     */
   public void windowClosing (WindowEvent e)
   {

        manager.setVisible (false);
        manager.dispose ();

      //System.exit (0);
   }

}
```

```
package caps.GraphEditor;

import javax.swing.*;
import java.awt.*;

import java.awt.geom.*;
import java.awt.event.*;
import java.awt.print.*;
import java.util.*;
import caps.Psdl.*;
import caps.Display.*;

//add SYT
import javax.swing.tree.*;

/**
  * The drawpanel is the place where the prototypes are
  * drawn on the screen and implements the undo, redo functionality.
  * and lot of user usability issues
  *
  * @author Shen-Yi Tao
  * @version 1.1
  */
public class  DrawPanel extends JPanel implements MouseListener
, MouseMotionListener, ActionListener,KeyListener
{
  /**
    * Store space for redo
    */
  protected Vector RedoVector;
  //add 1/2/00 SYT
  /**
    * Store space for undo
    */
  protected Vector UndoVector ;

  //add 1/28/00 SYT
  /**
    *identify if component is dragged
    */
  protected boolean draggedFlag;

  //add 1/29/00 SYT
  /**
    *identify if component is added
    */
  protected boolean addFlag;
  //add 1/29/00 SYT
  /**
    *identify if a stream is drawing
    */
  protected boolean isDrawingStream;

  //add 1/29/00 SYT
  /**
    *identify if a EXTERNAL is drawing
    */
  protected boolean isExternalDrawed;
  //add 1/30/00 SYT
  /**
    * is mouse pressed?
    */
  protected boolean isPressedOn;
  /**
    * The constant width of the DrawPanel
    */
  public static final int WIDTH = 1024;

    /**
```

171

```
    * The constant height of the DrawPanel
    */
public static final int HEIGHT = 768;

private final Cursor DEFAULT_CURSOR = new Cursor (Cursor.DEFAULT_CURSOR);

private final Cursor HAND_CURSOR = new Cursor (Cursor.HAND_CURSOR);

private final Cursor MOVE_CURSOR = new Cursor (Cursor.MOVE_CURSOR);

/**
  * The constant which specifies an operator
  */
public final static int OPERATOR = 1;

/**
  * The constant which specifies a terminator
  */
public final static int TERMINATOR = 2;

/**
  * The constant which specifies a stream
  */
public final static int STREAM = 3;

/**
  * The value of this variable is true if the toolbar is
  *in the select mode
  */
protected boolean selectMode;

/**
  * The frame which has created this DrawPanel object
  */
protected Editor parentFrame;

/**
  * This vector holds the shapes that are drawn in the DrawPanel.
  * Each shape is redrawn in the paint method by polling them
  * from this Vector.
  */
protected Vector displayComponentVector;
/**
  * vector for handle_marker on each component. SYT
  */
protected Vector handlesVector;
/**
 *this is only for single selection 1/6/00 SYT
 */
protected DisplayComponent selectedComponent;
//add 1/30/00 SYT
/**
 * current display external
 */
protected DisplayExternal currentExternal;

protected boolean MOVING_COMPONENT = false;

protected boolean MOVING_LABEL = false;

protected boolean MOVING_MET = false;

protected boolean RESIZING = false;

protected boolean IS_COLLECTING_POINTS = false;

protected boolean MOVING_ALL = false;

protected Point2D diagonalPoint;
```

172

```
protected VertexProperties vPropertyPanel;

protected EdgeProperties ePropertyPanel;

protected Vertex parentVertex; // The parent of the current Level

protected EdgePath currentEdge;

protected boolean selectionDefault;

/**
  * Current component is either an OPERATOR, or a TERMINATOR or a STREAM
  * according to the selection from the toolbar.
  */
protected int currentComponent;

protected Popup popupMenu;

protected boolean selectAllMode;

protected Point prevPoint;

protected Rectangle bounds;

protected int currentColor;

protected int currentFont;



Graphics osG;   //the off-screen Graphics object to draw on
Image osImg;    //an Image to link to the off-screen Graphics
boolean startOut; //is this the first time we're painting?
Dimension osD;            //the dimensions of the off-screen Graphics



/**
  * Constructs a new ToolBar object
  *
  * @param frame The parent frame of this DrawPanel object.
  */
public DrawPanel (Editor frame, Vertex root)
{
    super();

    //add 2/3/00 SYT
    requestFocus();

    startOut = true;

    //add  1/2/00 SYT
    // redo vector
    RedoVector=new Vector(1);

    //add  1/2/00 SYT
    UndoVector=new Vector(1);

    //add 1/29/00 SYT
    //is now drawing a stream?
    isDrawingStream=false;
    //add 1/28/00 SYT
    //is mouse dragged?
    draggedFlag = false;
    //add 1/29/00 SYT
    // is component added?
    addFlag = false;
    //add 1/29/00 SYT
```

```
    // is External drawed?
    isExternalDrawed=false;
    //add 1/30/00 SYT
    // current focused Enternal
    currentExternal=null;
    //add 1/30/00 SYT
    // is mouse pressed on component?
    isPressedOn =false;

    popupMenu = new Popup (this);

    parentFrame = frame;

    setAlignmentX (LEFT_ALIGNMENT); // Panel does not accept these
    setAlignmentY (TOP_ALIGNMENT);
    setBorder (BorderFactory.createEtchedBorder ());

    selectMode = true;   // Initially in the selectMode.

    selectedComponent = null;

    currentEdge = null;

    vPropertyPanel = new VertexProperties (frame);
    ePropertyPanel = new EdgeProperties (frame);
    vPropertyPanel.setVisible (false);
    ePropertyPanel.setVisible (false);

    TextEditor editor = new TextEditor (frame);
    IdListEditor idEditor = new IdListEditor (frame);
    //add 10/13/00 SYT
    OtherPropertiesEditor opEditor = new OtherPropertiesEditor(frame);

    displayComponentVector = new Vector ();
    handlesVector = new Vector ();

    parentVertex = root;       // This is the root

    diagonalPoint = null;

    setCursor (DEFAULT_CURSOR);

    selectAllMode = false;
    selectionDefault = false;

    prevPoint = new Point ();

    bounds = null;

    currentColor = 61;   // White; 61 because it is index
    currentFont = 4;     // Courier Plain 12  (it is 5 - 1 )

    addMouseListener (this);                  // Register mouse events
    addMouseMotionListener (this);
    addKeyListener(this);
}

//add 2/3/00 SYT
/**
 * is component selected
 */
public boolean isDFCSelected()
{
 if(selectedComponent == null)
 return false;
 else
 return true;
}
```

```java
//add 1/16/00 SYT
/**
 * push tree model to undo vector
 */
public void pushUndoVector()
{
    DefaultTreeModel DTM = parentFrame.getTreePanel().cloneTreeModel();

    UndoVector.addElement(DTM);
    parentFrame.getEditorMenuBar().getEditMenu().getUndoMenuItem()
    .setEnabled(true);
    parentFrame.getToolBar().getUndoButton().setEnabled(true);
}
 //add 1/16/00 SYT
/**
 * push tree model to redo vector
 */
public void pushRedoVector()
{
    DefaultTreeModel DTM = parentFrame.getTreePanel().cloneTreeModel();
    RedoVector.addElement(DTM);
    parentFrame.getEditorMenuBar().getEditMenu().getRedoMenuItem()
    .setEnabled(true);
    parentFrame.getToolBar().getRedoButton().setEnabled(true);
}


//add SYT 1/16/00
/**
 * pop from undo vector
 */
public void popUndoVector()
{
    if(! UndoVector.isEmpty())
    {
      UndoVector.removeElement(UndoVector.lastElement());
    }
       //set undo to false. SYT
    if ( UndoVector.isEmpty() )
    {
      parentFrame.getEditorMenuBar().getEditMenu().getUndoMenuItem()
      .setEnabled(false);
      parentFrame.getToolBar().getUndoButton().setEnabled(false);
    }
}


//add SYT 1/16/00
/**
 * pop from redo vector
 */
public void popRedoVector()
{
    if(! RedoVector.isEmpty())
    {
      RedoVector.removeElement(RedoVector.lastElement());
    }

    //set undo to false. SYT
    if ( RedoVector.isEmpty() )
    {
       parentFrame.getEditorMenuBar().getEditMenu().getRedoMenuItem()
       .setEnabled(false);
       parentFrame.getToolBar().getRedoButton().setEnabled(false);
    }
}

//add 1/16/00 SYT
/**
```

```
 * used before setParentVertex()
 * update tree model
 */
public void updateTreePanel(DefaultTreeModel DTM)
{
    parentFrame.getTreePanel().updateTreeModel(DTM);
}


//add 2/9/00 SYT
/**
 * find the parent vertex in the generated clone tree model
 */
public Vertex findCloneParentVertex(DefaultTreeModel DTM)
{
    for(Enumeration e =( (DefaultMutableTreeNode )( DTM.getRoot()) )
      .breadthFirstEnumeration(); e.hasMoreElements() ; )
    {
      DefaultMutableTreeNode DMTN =
                           (DefaultMutableTreeNode )( e.nextElement() );
      if( (DMTN instanceof Vertex) && !(DMTN instanceof External) )
      {
        if( ( (Vertex)DMTN ).getIsParent() )
        {
            return (Vertex)DMTN;
        }
      }
    }
    return (Vertex)(DTM.getRoot());
}

//add. 1/16/00 SYT
/**
 * paint screen for undo
 */
public void undoPaint()
{
  //for debug  SYT
  if(UndoVector.isEmpty() )
  {
      System.out.println("Error! Undo Vector is empty.");
  }
  else
  {
      pushRedoVector() ;
      DefaultTreeModel model = (DefaultTreeModel)(UndoVector.lastElement());

      updateTreePanel( model );
      //update clone parentVertex SYT
      DefaultTreeModel currentModel = parentFrame.getTreePanel()
                                    .getTreeModel() ;
      parentVertex = findCloneParentVertex(currentModel);
      //test
      if(parentVertex == null)
      {
        System.out.println("parentVertex == null");
      }

      //add 2/15/00 SYT
      setTreePath();
      setParentVertex(parentVertex ,null);

      selectAllMode=false;
      //add 1/16/00 SYT
      setSelectMode (true);
      popUndoVector();
  }
  //set undo disable. SYT
  if(UndoVector.isEmpty())
  {
```

```
             parentFrame.getEditorMenuBar().getEditMenu()
             .getUndoMenuItem().setEnabled(false);

              parentFrame.getToolBar().getUndoButton().setEnabled(false);
    }
}

//add. 1/16/00 SYT
/**
 * paint screen for redo
 */
public void redoPaint()
{
   //for debug  SYT
   if(RedoVector.isEmpty() )
   {
      System.out.println("Error!Redo Vector is empty.");
   }
   else
   {
      pushUndoVector() ;
      //updateTreePanel( (DefaultTreeModel)(RedoVector.lastElement()) );
      DefaultTreeModel model = (DefaultTreeModel)(RedoVector.lastElement());
      updateTreePanel(model);
      //update clone parentVertex SYT
      DefaultTreeModel currentModel = parentFrame.getTreePanel()
                                       .getTreeModel() ;
      //update clone parentVertex SYT
      parentVertex = findCloneParentVertex(currentModel);

      //add 2/15/00 SYT
      setTreePath();

      setParentVertex(parentVertex ,null);

      selectAllMode=false;
      //add 1/16/00 SYT
      setSelectMode (true);
      popRedoVector();
   }
   //set redo disable. SYT
   if(RedoVector.isEmpty())
   {
      parentFrame.getEditorMenuBar().getEditMenu()
      .getRedoMenuItem().setEnabled(false);

       parentFrame.getToolBar().getRedoButton().setEnabled(false);
   }

}
//add 1/29/00 SYT
/**
 * get current selected edge.
 */
public EdgePath getCurrentEdge()
{
    return currentEdge;
}
// add 2/15/00 SYT
/**
 *set path to parent vertex
 */
public void setTreePath()
{
    DefaultMutableTreeNode selectedParent =
    ((DefaultMutableTreeNode)parentVertex);
    //set path of the selected component
    TreeNode [] obj= selectedParent.getPath();
    TreePath TP = new TreePath(obj);
```

177

```
      parentFrame.getTreePanel().setSelectionPath(TP);
      //ensure selected path visible
      parentFrame.getTreePanel().expandPath(TP);
}

 // modify 2/14/00 SYT
/**
  * Sets the select mode to true or false. The panel is generally in
  * the select mode unless another button is pressed in the toolbar.
  * Update tree panel
  * @param mode true if the panel is going to be in the select mode.
  */
public void setSelectMode (boolean mode)
{
    selectMode = mode;

    if (selectMode == false && selectedComponent != null)
    {
      selectedComponent = null;
      eraseHandles ();
    }
}
// 11/19/99 SYT
/**
 * go to root node
 */
public void gotoRoot ()
{
    //add 8/22/00 SYT
    //parentFrame.getToolBar ().setTerminatorButton(true);

    TreeNode [] obj;
    if (parentVertex.isRoot () == false)
    {
       setParentVertex ((Vertex) parentVertex.getRoot (), null);
       obj = ((DefaultMutableTreeNode)parentVertex.getRoot())
                         .getPath();
    }
    else
    {
      obj= ((DefaultMutableTreeNode)parentVertex).getPath();
    }

    TreePath TP = new TreePath(obj);
    parentFrame.getTreePanel().setSelectionPath(TP);
    //ensure selected path visible
    parentFrame.getTreePanel().expandPath(TP);

    parentFrame.getToolBar ().setOperatorButton (true);

}

 // redesign  2/22/00 SYT
/**
 * go to parent
 * this function is closed ; leaving for reference
 */
public void gotoParent ()
{
    if( displayComponentVector.isEmpty())
    {
      DefaultMutableTreeNode upperLevelParent = parentVertex ;
      // If this is not the root of this tree
      setParentVertex ((Vertex)upperLevelParent , null);
      parentFrame.getToolBar ().setOperatorButton (true);
      //set path of the selected component
      TreeNode [] obj= upperLevelParent.getPath();
      TreePath TP = new TreePath(obj);
      parentFrame.getTreePanel().setSelectionPath(TP);
```

```
      parentFrame.getTreePanel().expandPath(TP);
    }
    if(parentVertex.isRoot())
    {
      //set path of the selected component
      TreeNode [] obj= parentVertex.getPath();
      TreePath TP = new TreePath(obj);
      parentFrame.getTreePanel().setSelectionPath(TP);
      parentFrame.getTreePanel().expandPath(TP);
    }

    if (!parentVertex.isRoot()/*&&
        !((DefaultMutableTreeNode)(parentVertex.getParent())).isRoot()*/)
    {
      int colonVertexID = parentVertex.getCloneVertexID();

      DefaultMutableTreeNode upperLevelParent = (DefaultMutableTreeNode)
                                         (parentVertex.getParent() ) ;

       //handel on parentVertex  2/21/00 SYT

      // paint (getGraphics ());
      // If this is not the root of this tree  SYT
      setParentVertex ((Vertex)upperLevelParent , null);
      parentFrame.getToolBar ().setOperatorButton (true);
      //set path of the selected component  SYT
      TreeNode [] obj= upperLevelParent.getPath();
      TreePath TP = new TreePath(obj);
      parentFrame.getTreePanel().setSelectionPath(TP);
      parentFrame.getTreePanel().expandPath(TP);

    }
}

 //* redesign 3/7/00 SYT
/**
 * create a health decompose
 * call treepanel valuechange() event
 * to make a decompse
 */
public void decompose ()
{
    //1/6/00 SYT
    if (selectedComponent==null)
       return;

    //set path of the selected component
    TreeNode [] obj=
    ((DefaultMutableTreeNode)selectedComponent.getDataFlowComponent())
     .getPath();
    TreePath TP = new TreePath(obj);
    parentFrame.getTreePanel().setSelectionPath(TP);
    //ensure selected path visible
    parentFrame.getTreePanel().expandPath(TP);

}
//redesign 8/22/00 SYT
/**
 * Invoked from the treepanel
 * Invoked when the forcused node on the treePanel has been changed.
 * @ parent- parent vertex
 */
public void changeLevel (Vertex parent)
{
    setParentVertex (parent, null);
    if (parent.isTerminator ())
      parentFrame.getToolBar ().setOperatorButton (false);

    else
```

```
            parentFrame.getToolBar ().setOperatorButton (true);
}
// redesign 2/21/00 SYT
/**
 * go to one upper level screen
 * @param v :upper level parent vertex
 * @param g2d : Graphics2D Object for drawing current screen.
 */
public void setParentVertex (Vertex v, Graphics2D g2D)
{
   //reuse this if need a gotoParent button  close at 2/28/00 SYT
   //add 2/21/00 SYT
  /* if(v.isRoot() | selectedComponent != null)
   {
     parentFrame.getToolBar().getGoToParentButton().setEnabled(false);
   }
   else
   {
     parentFrame.getToolBar().getGoToParentButton().setEnabled(true);
   }*/
   //add 3/10/00 SYT
   //reset the current parent Vertex on this screen
   // resetParentVertex(v);

   setSelectMode (false);
   parentVertex = v;

   if (g2D == null)
       g2D = (Graphics2D) getGraphics ();

   displayComponentVector.removeAllElements ();

   for (Enumeration enum = parentVertex.children ()
                            ; enum.hasMoreElements ();)
   {
      DataFlowComponent dfc = (DataFlowComponent) enum.nextElement ();
      if (dfc instanceof Vertex && !(((Vertex) dfc).isTerminator ()) &&
          !(dfc instanceof External))
      {
         DisplayVertex op = new DisplayVertex ((Vertex) dfc);
         op.setLabelShape (g2D);
         op.setMetShape (g2D);
         //add element
         displayComponentVector.addElement (op);
      }
      else if (dfc instanceof Vertex && (((Vertex) dfc).isTerminator ())
             && !(dfc instanceof External))
      {
         DisplayVertex tr = new DisplayVertex ((Vertex) dfc);
         tr.setLabelShape (g2D);
         tr.setMetShape (g2D);
         displayComponentVector.addElement (tr);
      }
      else
      {
         EdgePath ep = new EdgePath ((Edge) dfc);
         ep.setLabelShape (g2D);
         ep.setMetShape (g2D);
         ((Edge) ep.getDataFlowComponent ()).correctLabelOffset ();

         ep.update ();
         displayComponentVector.addElement (ep);

         if (((Edge) dfc).getSource () instanceof External)
         {
            DisplayExternal extern =
                 new DisplayExternal ((External)((Edge) dfc).getSource ());
            extern.setLabelShape ((Graphics2D) getGraphics ());
            extern.setMetShape ((Graphics2D) getGraphics ());
```

180

```java
                displayComponentVector.addElement (extern);
            }
            else if (((Edge) dfc).getDestination () instanceof External)
            {
                DisplayExternal extern =
                  new DisplayExternal ((External)((Edge) dfc).getDestination ());
                extern.setLabelShape ((Graphics2D) getGraphics ());
                extern.setMetShape ((Graphics2D) getGraphics ());
                displayComponentVector.addElement (extern);
            }
        }
    }

    //clearAllComponentsFromScreen (g2D);
    paint (g2D);
    setSelectMode (true);
}

//modify 2/14/00 SYT
/**
 * erase the selected maker
 */
public void eraseHandles ()
{
    handlesVector.removeAllElements ();
    //clearAllComponentsFromScreen (null);
    paint (getGraphics ());
}
 //add SYT 12/31/99
/**
 * clearAllComponent usefull for only refresh this screen.
 */
public void clearAllComponentsFromScreen (Graphics2D g2D)
{
    if (g2D == null)
        g2D = (Graphics2D) getGraphics ();
    g2D.setColor (Color.white);
    g2D.fillRect (0, 0, WIDTH, HEIGHT);
}

/**
  * Sets the currentComponent variable to the specified argument.
  * @param component  OPERATOR, TERMINATOR or STREAM
  */
public void setCurrentComponent (int component)
{
    currentComponent = component;
}

//redesign 1/24/00 SYT
/**
 * mark the selected DFC and notify the tree panel
 * @param dfc selected component
 */
public void setSelectedDFC (DataFlowComponent dfc)
{
    DisplayComponent dc;
    for (Enumeration enum = displayComponentVector.elements ()
                                            ; enum.hasMoreElements ();)
    {
        dc = (DisplayComponent) enum.nextElement ();
        //find the selcted dfc from the display vector and mark it.
        if (dc.getDataFlowComponent ().equals (dfc))
        {
            selectedComponent = dc;
            handlesVector = dc.getHandles ();
            paint (getGraphics ());
        }
    }
```

181

```
      }
       //SYT
      /**
        * Creates a new Operator and a new OperatorCircle object.
        * Calls the paintComponent () method to draw the component to this panel.
        * @param xLoc The x location of the component.
        * @param yLoc The y location of the component.
        */
      public void processOperator (int xLoc, int yLoc)
      {
         Graphics2D g2D = (Graphics2D) getGraphics ();
         if (selectionDefault)
            setSelectMode (true); // It will allow to place only one component
                                  // at a time  SYT
         //generate new Vertex. SYT
         Vertex op = new Vertex (xLoc, yLoc, parentVertex, false);
         op.setColor (currentColor + 1);
         op.setLabelFontIndex (currentFont + 1);
         op.setMetFontIndex (currentFont + 1);
         //add to tree. 1/9/00 SYT
         parentFrame.getTreePanel().addNewDFC (op, parentVertex);
         DisplayVertex opCircle = new DisplayVertex (op);
         opCircle.setLabelShape (g2D);
         opCircle.setMetShape (g2D);
         paintComponent (opCircle);
         displayComponentVector.addElement (opCircle);
      }

      public void processTerminator (int xLoc, int yLoc)
      {
         Graphics2D g2D = (Graphics2D) getGraphics ();
         if (selectionDefault)
            setSelectMode (true);  // It will allow to place only one component
                                   //at a time
         Vertex term = new Vertex (xLoc, yLoc, parentVertex, true);
         term.setColor (currentColor + 1);
         term.setLabelFontIndex (currentFont + 1);
         term.setMetFontIndex (currentFont + 1);
         //add to tree. 1/9/00 SYT
         parentFrame.getTreePanel ().addNewDFC (term, parentVertex);
         DisplayVertex tRectangle = new DisplayVertex (term);
         tRectangle.setLabelShape (g2D);
         tRectangle.setMetShape (g2D);
         paintComponent (tRectangle);
         displayComponentVector.addElement (tRectangle);
      }
      //SYT
      /**
        * errors and too simple ,redesign is needed.
        * first find source/destination
        * check if this is the second or more click
        */
      public void processStream (int x, int y, int clicks)
      {
         isDrawingStream = true;
         DisplayComponent dc;
         Vertex v = null;
         //find destination vertex SYT
         for (Enumeration enum = displayComponentVector.elements (); enum.hasMoreElements
();)
         {
            dc = (DisplayComponent) enum.nextElement ();

            if (dc instanceof DisplayVertex && dc.getShape ().contains (x, y))
               v = (Vertex) dc.getDataFlowComponent ();
         }
         //if statement has been modifed 1/29/00 SYT
         if (IS_COLLECTING_POINTS) // Second or more click   SYT
         {
```

```
            //find the destination SYT
            if (v != null)
            {
               //add 2/3/00 SYT
               //clearn the stream trace on the screen
               currentEdge.reset();
               // Found the destination
               v.addInEdge ((Edge) currentEdge.getDataFlowComponent ());
               ((Edge) currentEdge.getDataFlowComponent ()).setDestination (v);
               ((Edge) currentEdge.getDataFlowComponent ()).addPoint (x, y);
               parentFrame.getTreePanel ().addNewDFC ((Edge)
currentEdge.getDataFlowComponent (), parentVertex);
               currentEdge.setLabelShape ((Graphics2D) getGraphics ());
               currentEdge.setMetShape ((Graphics2D) getGraphics ());
               ((Edge) currentEdge.getDataFlowComponent ()).correctLabelOffset ();
               currentEdge.update ();
               IS_COLLECTING_POINTS = false;
               //add EdgePath to the display vector.  SYT
               displayComponentVector.addElement (currentEdge);
               if (selectionDefault)
               {
                  parentFrame.getToolBar ().enableSelectButton ();
                  setSelectMode (true);
               }
               //add 1/29/00 SYT
               //finish drawing
               isDrawingStream=false ;
               //add 1/30/00 SYT
               isExternalDrawed=false;
               //change 2/4/00 SYT
             // paintComponent (currentEdge);
//clearAllComponentsFromScreen (null);
               paint(getGraphics());
            }
            else
            {  // collect the next point
               ((Edge) currentEdge.getDataFlowComponent ()).addPoint (x, y);
               //add 2/3/00 SYT
               //draw line SYT
               currentEdge.quadTo(x,y);
               drawSegment(currentEdge.getShape());
            }
         }
      //1/29/00  SYT
      //stream source:external
      else if (v == null)
      {    // This is an external -> vertex stream
         //add 1/29/00 SYT
         pushUndoVector();
         //add 1/29/00 SYT
         isExternalDrawed=true;
         IS_COLLECTING_POINTS = true;
         //create new external  SYT
         External ex = new External (x, y, parentVertex);
         //create new edge  SYT
         Edge ed = new Edge (x, y, parentVertex);

         ed.setLabelFontIndex (currentFont + 1);
         ed.setMetFontIndex (currentFont + 1);
         ex.setLabelFontIndex (currentFont + 1);
         ex.setMetFontIndex (currentFont + 1);
         //add out edge SYT
         ex.addOutEdge (ed);
         //set destination SYT
         ed.setSource (ex);
         //create edge path SYT
         currentEdge = new EdgePath (ed);

         //add 2/3/00 SYT
```

```
            //draw segment
            currentEdge.quadTo(x,y);
            drawSegment(currentEdge.getShape());

            //create display extern  SYT
            DisplayExternal extern = new DisplayExternal (ex);
            //add 1/30/00 SYT
            currentExternal=extern;
            extern.setLabelShape ((Graphics2D) getGraphics ());
            extern.setMetShape ((Graphics2D) getGraphics ());          // *** Maybe we don't
need this ******
            //display component   SYT
            displayComponentVector.addElement (currentExternal);
            paintComponent (currentExternal);
        }
        //first clicled SYT
        // source : vertex
        else
        {
            //add 1/29/00 SYT
            pushUndoVector();
            // First click         //  vertex-vertex or vertex-external
            IS_COLLECTING_POINTS = true;
            Edge ed = new Edge (x, y, parentVertex);

            ed.setLabelFontIndex (currentFont + 1);
            ed.setMetFontIndex (currentFont + 1);
            v.addOutEdge (ed);
            ed.setSource (v);
            currentEdge = new EdgePath (ed);
            //add 2/3/00 SYT
            //draw segment
            currentEdge.quadTo(x,y);
            drawSegment(currentEdge.getShape());
        }
     }
    // add 2/3/00 SYT
    /**
     * draw segment between two points
     */
     public void drawSegment(Shape gp)
     {
        Shape shape = gp;

        Graphics2D g2D = (Graphics2D) getGraphics ();
     //  g2D.setColor (new Color (ColorConstants.RGB_VALUES [currentColor]));
        g2D.fill (shape);
        g2D.setColor (Color.blue);
        g2D.draw (shape);
     }


    /**
      * Paints the component into this panel by calling the graphics2D.draw(Shape) method.
      * @param component The component to be drawn into the panel
      */
    public void paintComponent (DisplayComponent component)
    {
        Graphics2D g2D = (Graphics2D) getGraphics ();

        if (component instanceof DisplayVertex)
        {
            g2D.setColor (new Color (ColorConstants.RGB_VALUES [currentColor]));
            //g2D.fill (((DisplayVertex) component).getPaintedShape ());
            g2D.fill (component.getShape ());
            g2D.setColor (Color.black);
        }
        g2D.draw (component.getShape ());
        component.drawLabelShape (g2D);
```

184

```java
        if (component instanceof DisplayVertex)
           component.drawMetShape (g2D);
   }

   /**
     * This method is called to repaint
     * all the components when necessary.
     * @param g The graphics context of the panel
     */
   public void paint (Graphics g)
   {
//System.out.println("Paint called");
//System.out.println(startOut);
       //*********************double-buffer*******************************
       //added 02/08/01 to fix problems with drawPanel rendering within the split panel
       if (startOut) {

//System.out.println("Entered first if statement");

           //startOut = false;

//System.out.println("Calling osD = getSize();");

           osD = getSize();

//System.out.println("Calling osImg = createImage()");

           osImg = createImage ((int)osD.getWidth(), (int)osD.getHeight());

//System.out.println("Calling osG = osImg.getGraphics()");

           osG = osImg.getGraphics();

           startOut = false;
       }
       if ((osD.getWidth() != getSize().getWidth()) || (osD.getHeight() !=
getSize().getHeight())) {
           osD = getSize();
           osImg = createImage ((int)osD.getWidth(), (int)osD.getHeight());
           osG = osImg.getGraphics();
       }
//System.out.println("Calling osG.setColor()");

       osG.setColor (Color.white);

//System.out.println("Calling osG.fillRect()");

       osG.fillRect (0,0,osD.width, osD.height);

       //*********************double-buffer*******************************


     //modified 2/15/00 SYT
      Graphics2D g2D;

      if(osG == null)//changed g to osG as part of double-buffer
      {
        g2D = (Graphics2D) getGraphics();
      }
      else
      {
        g2D = (Graphics2D) osG;//changed g to osG as part of double-buffer
      }
      g2D.setColor (Color.black);

      for (Enumeration e = displayComponentVector.elements ()
                                      ; e.hasMoreElements ();)
      {
         DisplayComponent dcp = (DisplayComponent) e.nextElement ();
```

185

```
        DataFlowComponent dfc = dcp.getDataFlowComponent ();
        if (MOVING_COMPONENT
            || (MOVING_LABEL && selectedComponent.getDataFlowComponent ()
                instanceof External))
            dcp.update ();          // and also label changes vs
        if (dfc instanceof Edge && ((Edge) dfc).isStateStream ())
        {
            g2D.setStroke (new BasicStroke (1.5f));
            g2D.draw (dcp.getShape ());
            g2D.setStroke (new BasicStroke (1f));
        }
        else
        {
            if (dcp instanceof DisplayVertex)
            {
                g2D.setColor (new Color (ColorConstants.RGB_VALUES [((Vertex)
dfc).getColor () - 1]));
                g2D.fill (((DisplayVertex) dcp).getPaintedShape ());
                g2D.setColor (Color.black);
            }
            g2D.draw (dcp.getShape ());
            if (!dfc.isLeaf ())
                g2D.draw (((DisplayVertex) dcp).getInnerShape ());
        }
        dcp.drawLabelShape (g2D);
        dcp.drawMetShape (g2D);
    }
    if ((selectMode && selectedComponent != null) || selectAllMode)
    {
        //mark the component. SYT 12/31/99
        for (Enumeration e = handlesVector.elements (); e.hasMoreElements ();)
        {
            g2D.setColor (Color.gray);
            g2D.fill ((Shape) e.nextElement ());
        }
    }

    g.drawImage(osImg,0,0,this);//changed g to osG
}

/**
 * Sets the size of the panel to WIDTH and HEIGHT
 * @return Returns a new Dimension object initialized to the
 *         WIDTH and HEIGHT parameters.
 */
public Dimension getPreferredSize ()
{
    return new Dimension (WIDTH, HEIGHT);
}

// redesign  SYT
/**
 * Handles the event that occurs when a mouse button is clicked on this panel
 * @param e The MouseEvent that occurs.
 */
public void mousePressed (MouseEvent e)
{
    int xPosition = e.getX ();
    int yPosition = e.getY ();

    int flags = e.getModifiers ();
    prevPoint.setLocation (xPosition, yPosition);

    //add SYT 12/31/99
    if ( e.isControlDown() )
    {
        //hot key not implemented yet. SYT
        //System.out.println("ControlDown");
    }
```

186

```
        //left mouse button or right mouse button was pressed. SYT 12/28/99
        else if (flags == MouseEvent.BUTTON1_MASK
                || flags == MouseEvent.BUTTON3_MASK)
        {

          if (!selectAllMode && isHoldingHandle
           (xPosition, yPosition))
          {
             //set RESIZING.(draggable)  1/12/00 SYT
            if (selectedComponent instanceof DisplayVertex &&
               selectedComponent.getShape ().getBounds2D ()
               .contains (diagonalPoint))  // Make sure it is not the label
               //risize component SYT
               RESIZING = true;
          }
          else if (selectMode)
          {
            DisplayComponent dc;
            boolean flag = false;
            for (Enumeration enum = displayComponentVector.elements ()
                 ; enum.hasMoreElements ();)
            {
              dc = (DisplayComponent) enum.nextElement ();
              if  (selectAllMode
                  && (dc.containsClickedPoint (xPosition, yPosition)
                 || dc.getLabelShapeBounds ().contains (xPosition, yPosition)
                 ||(dc.getMetShapeBounds ().contains (xPosition, yPosition))))
              {
                 MOVING_ALL = true;
                 flag = true;
              }
              //paint selected DataFLowComponent. 1/12/00 SYT
              else if (dc.getLabelShapeBounds ()
                                          .contains (xPosition, yPosition))
            {  // If clicked a label
                 eraseHandles ();

                 //add 2/4/00 SYT
                 if( !(dc.getDataFlowComponent() instanceof External))
                 {
                     //add 2/28/00 SYT
                     //Both of the parent and child forcused
                     //on the parent itself
                     TreeNode [] obj=
                     ((DefaultMutableTreeNode) ((DefaultMutableTreeNode)
                         (dc.getDataFlowComponent())).getParent()).getPath();
                     //add 2/12/00 SYT
                     TreePath TP = new TreePath(obj);
                     //add 2/12/00 SYT
                     //ensure selected path visible
                     parentFrame.getTreePanel().expandPath(TP);

                 }
                 //if(parentFrame.getTreePanel().isPathSelected(TP))
                 //parentFrame.getTreePanel().getTreeModel().reload();

                 //reset marker  SYT
                 handlesVector = dc.getStringHandles(dc.getLabelShapeBounds());
                 selectedComponent = dc;
//clearAllComponentsFromScreen((Graphics2D) getGraphics());
                 paint (getGraphics ());
                 flag = true;

             }
             else if (dc.getMetShapeBounds ().contains (xPosition, yPosition))
             { // If clicked an met
                 eraseHandles ();

                 //add 2/4/00 SYT
```

187

```
     if( !(dc.getDataFlowComponent() instanceof External) )
     {
       //add 2/15/00 SYT
       /*if(dc.getDataFlowComponent() instanceof Vertex)
       {
         parentFrame.getToolBar().getDecomposeButton()
         .setEnabled(true);
       }*/

       //changed on 2/28/00 SYT
       /*if( ((DefaultMutableTreeNode) (dc.getDataFlowComponent()))
         .isLeaf()  )
       {
         //set path of the selected component
         TreeNode [] obj= ((DefaultMutableTreeNode)
         (dc.getDataFlowComponent()) ).getPath();
         //add 2/12/00 SYT
         //following lines will make labe marker disapear
         TreePath TP = new TreePath(obj);
         parentFrame.getTreePanel().setSelectionPath(TP);
         //add 2/12/00 SYT
         //ensure selected path visible
         parentFrame.getTreePanel().expandPath(TP);
       }
       // if parent cancel the focuse
       else
       {
         //set path of the selected component
         TreeNode [] obj=
         ((DefaultMutableTreeNode) ((DefaultMutableTreeNode)
             (dc.getDataFlowComponent())).getParent()).getPath();
         //add 2/12/00 SYT
         TreePath TP = new TreePath(obj);
         //add 2/12/00 SYT
         //ensure selected path visible
         parentFrame.getTreePanel().expandPath(TP);

          parentFrame.getTreePanel().clearSelection();
       } */
       //add 2/28/00 SYT
        //Both of the parent and child forcused
        //on the parent itself
        TreeNode [] obj=
        ((DefaultMutableTreeNode) ((DefaultMutableTreeNode)
            (dc.getDataFlowComponent())).getParent()).getPath();
        //add 2/12/00 SYT
        TreePath TP = new TreePath(obj);
        //add 2/12/00 SYT
        //ensure selected path visible
        parentFrame.getTreePanel().expandPath(TP);
     }

     //if(parentFrame.getTreePanel().isPathSelected(TP))
     //parentFrame.getTreePanel().getTreeModel().reload();
     //reset marker  SYT
     handlesVector = dc.getStringHandles (dc.getMetShapeBounds ());

     selectedComponent = dc;
     paint (getGraphics ());
     flag = true;
   }
   else if (dc.containsClickedPoint (xPosition, yPosition))
   {  // If clicked on a component
     eraseHandles ();
   //add 2/4/00 SYT
    if( !(dc.getDataFlowComponent() instanceof External )  )
    {
      //add 2/15/00 SYT
      /* if(dc.getDataFlowComponent() instanceof Vertex)
```

188

```
                       {
                         parentFrame.getToolBar().getDecomposeButton()
                         .setEnabled(true);
                       }*/

                       /* change on 2/28/00 SYT
                       if( ((DefaultMutableTreeNode) (dc.getDataFlowComponent()) )
                         .isLeaf()  )
                       {
                         //set path of the selected component
                         TreeNode [] obj= ((DefaultMutableTreeNode)
                         (dc.getDataFlowComponent()) ).getPath();
                         //add 2/12/00 SYT
                         //following lines will make labe marker disapear
                         TreePath TP = new TreePath(obj);
                         parentFrame.getTreePanel().setSelectionPath(TP);
                         //add 2/12/00 SYT
                         //ensure selected path visible
                         parentFrame.getTreePanel().expandPath(TP);
                       }
                       // if parent cancel the focuse
                       else
                       {
                         //set path of the selected component
                         TreeNode [] obj=
                         ((DefaultMutableTreeNode) ((DefaultMutableTreeNode)
                              (dc.getDataFlowComponent())).getParent()).getPath();
                         //add 2/12/00 SYT
                         TreePath TP = new TreePath(obj);
                         parentFrame.getTreePanel().setSelectionPath(TP);
                         //add 2/12/00 SYT
                         //ensure selected path visible
                         parentFrame.getTreePanel().expandPath(TP);

                         parentFrame.getTreePanel().clearSelection();
                       }*/
                        //add 2/28/00 SYT
                        //Both of the parent and child forcused
                        //on the parent itself
                        TreeNode [] obj=
                        ((DefaultMutableTreeNode) ((DefaultMutableTreeNode)
                             (dc.getDataFlowComponent())).getParent()).getPath();
                        //add 2/12/00 SYT
                        TreePath TP = new TreePath(obj);
                        //add 2/12/00 SYT
                        //ensure selected path visible
                        parentFrame.getTreePanel().expandPath(TP);

                   }
                   //if(parentFrame.getTreePanel().isPathSelected(TP))
                   //parentFrame.getTreePanel().getTreeModel().reload();

                   //reset marker  SYT
                   eraseHandles();
                   handlesVector = dc.getHandles ();
                   selectedComponent = dc;
                   paint (getGraphics ());
                   flag = true;  // Clicked on a component
               }
               if (flag)
                   setCursor (MOVE_CURSOR);
           }
//add 2/14/00 SYT
//mouse press on empty space
if(!flag)
{
   setTreePath();
   //add 2/15/00 SYT
   /*
```

```java
                 parentFrame.getToolBar().getDecomposeButton().setEnabled(false);
                 */
                // parentFrame.getToolBar().getGoToParentButton()
                // .setEnabled(false);
            }

            if (selectedComponent != null && !flag)
            {
                selectedComponent = null;
                eraseHandles ();
                setSelectAllMode (false);
            }

            //add 1/28/00 SYT
            //popUndoVector may be excuted on mouseReleased
            if(flag|RESIZING)
            {
              pushUndoVector();
              //add SYT 1/30/00 SYT
              isPressedOn =true;
            }
            setMenuBarItems ();
        }

        //add component. SYT 1/12/00
        else if (flags != MouseEvent.BUTTON3_MASK)
        {
            parentVertex.setAllowsChildren (true);
            //switch  OPERATOR:  TERMINATOR:   STREAM:

            //add 1/29/00 SYT
            addFlag = true;

            switch (currentComponent)
            {
                case OPERATOR:
                    //add 1/29/00 SYT
                    pushUndoVector();

                    processOperator (xPosition, yPosition);
                    if (selectionDefault)
                        parentFrame.getToolBar ().enableSelectButton ();
                    parentFrame.setSaveRequired (true);
                    break;
                //error and has been changed to correct. SYT 1/12/00
                //case OPERATOR:
                case TERMINATOR:
                    //add 1/29/00 SYT
                    pushUndoVector();
                    processTerminator (xPosition, yPosition);
                    if (selectionDefault)
                        parentFrame.getToolBar ().enableSelectButton ();
                    parentFrame.setSaveRequired (true);
                    break;
                case STREAM:
                    //here,undo need be implemented in processStream() 1/29/00 SYT
                    //pushUndoVector();
                    processStream (xPosition, yPosition, e.getClickCount ()); // Pending
same as chriss' implementation
                    parentFrame.setSaveRequired (true);
                    break;
                default:
                    break;
            }
        }
         //add 1/30/00 SYT
        else
        isPressedOn =false;
    }
```

190

```
      //reuse this if need a gotoParent button  close at 2/28/00 SYT
      /* if(parentVertex.isRoot() | selectedComponent!=null )
      {
        parentFrame.getToolBar().getGoToParentButton().setEnabled(false);
      }
      else
      {
        parentFrame.getToolBar().getGoToParentButton().setEnabled(true);
      }*/

}
 // modify 4/19/00 SYT
 /**
  * set MenuItems
  * prevent type name Vertex from decomposing
  */
 public void setMenuBarItems ()
 {
    if (selectedComponent == null)
    {
       // delete SYT
       parentFrame.getJMenuBar ().getMenu (1).getItem (4).setEnabled (false);
       // decompose SYT
       //parentFrame.getJMenuBar ().getMenu (3).getItem (2).setEnabled (false);
       parentFrame.getJMenuBar ().getMenu (3).getItem (1).setEnabled (false);
        // color SYT
       parentFrame.getJMenuBar ().getMenu (2).getItem (0).setEnabled (true);
    }
    else
    { // delete
       parentFrame.getJMenuBar ().getMenu (1).getItem (4).setEnabled (true);
       if (selectedComponent instanceof EdgePath)
       {  // color
          parentFrame.getJMenuBar ().getMenu (2).getItem (0)
                                        .setEnabled (false);
          // decompose
          //parentFrame.getJMenuBar ().getMenu (3).getItem (2)
          //                              .setEnabled (false);
            parentFrame.getJMenuBar ().getMenu (3).getItem (1)
                                        .setEnabled (false);
       }
       else
       {  // color
          parentFrame.getJMenuBar ().getMenu (2).getItem (0).setEnabled(true);
           if (!(selectedComponent instanceof EdgePath))
           {
             if (((Vertex)(selectedComponent.getDataFlowComponent()) )
                                        .getLabel().indexOf(".")== -1)
                // decompose
                //parentFrame.getJMenuBar ().getMenu (3).getItem (2)
                //                              .setEnabled (true);
                parentFrame.getJMenuBar ().getMenu (3).getItem (1)
                                        .setEnabled (true);
            else
              // decompose
              //parentFrame.getJMenuBar ().getMenu (3).getItem (2)
              //                              .setEnabled (false);
             parentFrame.getJMenuBar ().getMenu (3).getItem (1)
                                        .setEnabled (true);
          }
       }
    }
 }

 /**
   * Handles the event that occurs when the mouse enters into the panel.
   *
   * @param e The MouseEvent that occurs.
   */
```

191

```
    public void mouseEntered (MouseEvent e)
    {
    }

   /**
     * add 1/30/00 SYT
     * Handles the event that occurs when the mouse exits the panel.
     * dress stream_drawing_unfinish problem
     * @param e The MouseEvent that occurs.
     */
   public void mouseExited (MouseEvent e)
   {
      clearnDrawStream();
   }

   /**
    * Add 2/3/00 SYT
    * deal with unfinished job after stop drawing stream
    * clean screen
    */
    public void clearnDrawStream()
    {
        if(!(currentEdge==null) )
        {
           if(isDrawingStream)
           {
             //external->vertex SYT
             if(isExternalDrawed)
             {
                 //delete external and it's children
               // currentEdge.delete();
              int index=parentVertex.getIndex(currentEdge.getEdge());
              ((DefaultMutableTreeNode)parentVertex).remove(index);

               displayComponentVector
               .removeElement(currentExternal);
               parentFrame.getTreePanel ().removeDfc();

               //setParentVertex (parentVertex, null);
// clearAllComponentsFromScreen (null);
               paint(getGraphics());
             }
             //vertex->    SYT
             else
             {
              //remove edge  SYT

              int index=parentVertex.getIndex(currentEdge.getEdge());
             //add 2/9/00 SYT
              currentEdge.getEdge().getSource().removeOutEdge(currentEdge.getEdge());
             ((DefaultMutableTreeNode)parentVertex).remove(index);

              parentFrame.getTreePanel ().removeDfc();

 //clearAllComponentsFromScreen (null);
              paint(getGraphics());
             }

              if (selectionDefault)
              {
                 parentFrame.getToolBar ().enableSelectButton ();
                 setSelectMode (true);
              }
              //still focus on draw stream.  SYT
              setSelectMode (false);
              setCurrentComponent (DrawPanel.STREAM);

              popUndoVector();
              isExternalDrawed=false;
```

```
                isDrawingStream=false;
                currentEdge=null;
                currentExternal=null;
                IS_COLLECTING_POINTS=false;
            }
        }
    }
    //modify  4/19/00 SYT
  /**
    * Handles the event that occurs when a mouse button is pressed on this panel
    * prevent type-name vertex from decomposing
    * @param e The MouseEvent that occurs.
    */
  public void mouseClicked(MouseEvent e)
  {
      int xPosition = e.getX ();
      int yPosition = e.getY ();
      int flags = e.getModifiers ();
      int clickCount = e.getClickCount ();
      //non-External DataFlowComponent selected and not in "select all mode".  SYT
1/12/00
      if (selectedComponent != null && !(selectedComponent.getDataFlowComponent ()
instanceof External)
          && !selectAllMode)
      {
          //right mouse button clicked.
          //pop up a menu.
          //SYT 12/28/99
          if (flags == MouseEvent.BUTTON3_MASK)
            if (selectedComponent instanceof EdgePath)
                popupMenu.showPopupMenu (true, xPosition, yPosition);  // disables
decompose
            else
            {
                if (((Vertex)(selectedComponent.getDataFlowComponent())
).getLabel().indexOf(".")== -1)
                {
                    popupMenu.showPopupMenu (false, xPosition, yPosition);  // enables
decompose
                }
                else
                {
                     // disables decompose
                    popupMenu.showPopupMenu (true, xPosition, yPosition);
                }
            }
        else if (clickCount > 1)
            showProperties (selectedComponent);
      }
      //vertex->external SYT
      else if (flags == MouseEvent.BUTTON3_MASK && IS_COLLECTING_POINTS)
      {
          //add 4/20/00 SYT
          if( !(((Edge)currentEdge.getDataFlowComponent())
            .getSource() instanceof External) )
          {
           //add 2/3/00 SYT
           //clearn the stream trace on the screen
           currentEdge.reset();
           //drawing a stream destination is External  SYT
           External ex = new External (xPosition, yPosition, parentVertex);
           //external add in_edge  SYT
           ex.addInEdge ((Edge) currentEdge.getDataFlowComponent ());
           //set destination     SYT
           ((Edge) currentEdge.getDataFlowComponent ()).setDestination (ex);
           ((Edge) currentEdge.getDataFlowComponent ())
                              .addPoint (xPosition, yPosition);
           parentFrame.getTreePanel ().addNewDFC ((Edge) currentEdge
                              .getDataFlowComponent (), parentVertex);
```

193

```java
          currentEdge.setLabelShape ((Graphics2D) getGraphics ());
          currentEdge.setMetShape ((Graphics2D) getGraphics ());
          ((Edge) currentEdge.getDataFlowComponent ()).correctLabelOffset ();
          currentEdge.update ();
          IS_COLLECTING_POINTS = false;
          displayComponentVector.addElement (currentEdge);
          if (selectionDefault)
          {
            parentFrame.getToolBar ().enableSelectButton ();
            setSelectMode (true);
          }
          DisplayExternal extern = new DisplayExternal (ex);
          extern.setLabelShape ((Graphics2D) getGraphics ());
          extern.setMetShape ((Graphics2D) getGraphics ());
          displayComponentVector.addElement (extern);
          //change 2/3/00 SYT
          //paintComponent (currentEdge);
          //paintComponent (extern);
//clearAllComponentsFromScreen (null);
          paint(getGraphics());

          //add 2/2/00 SYT
          isExternalDrawed=false;
          isDrawingStream=false;
          currentEdge=null;
          currentExternal=null;
        }
      }
    }

    public void showProperties (DisplayComponent d)
    {
      if (d instanceof DisplayVertex)
      {
        vPropertyPanel.setVertex ((Vertex) d.getDataFlowComponent ());
        vPropertyPanel.setDisplayVertex ((DisplayVertex) d);
      }
      else
      {
        ePropertyPanel.setEdge ((Edge) d.getDataFlowComponent ());
        ePropertyPanel.setEdgePath ((EdgePath) d);
      }
    }

    /**
     * Handles the event that occurs when a mouse button is released
     * on this panel
     *
     * @param e The MouseEvent that occurs.
     */
    public void mouseReleased (MouseEvent e)
    {
        //add 1/26/00 SYT
      if( (currentComponent != STREAM)  )
      {
        if(isPressedOn)
        {
          //if(!(draggedFlag |addFlag) & !selectMode)
          if(draggedFlag |addFlag)
          {
          }
          else
          {
            popUndoVector();
          }
        }
      }
      //add 1/29/00 SYT
      isPressedOn=false;
```

194

```
      draggedFlag=false;
      addFlag=false;

      //add 1/28/00 SYT
      MOVING_COMPONENT = false;
      MOVING_LABEL = false;
      MOVING_MET = false;
      RESIZING = false;
      MOVING_ALL = false;
      diagonalPoint = null;
      //add 2/28/00
      setSelectAllMode(false);

   //setCursor (DEFAULT_CURSOR);
}


/**
  * Handles the event that occurs when the mouse is dragged on this panel
  *
  * @param e The MouseEvent that occurs.
  */
public void mouseDragged (MouseEvent e)
{
   if (selectedComponent != null || MOVING_ALL)
   {
      int xPosition = e.getX ();
      int yPosition = e.getY ();
      int flags = e.getModifiers ();

      if (!MOVING_ALL)
      {
         bounds = (Rectangle) selectedComponent.getShape ().getBounds ();
      }
      if (((bounds.getMinX () <= 0) && (xPosition <= prevPoint.x)) ||
          ((bounds.getMinY () <= 0) && (yPosition <= prevPoint.y)) ||
          ((bounds.getMaxX () >= WIDTH) && (xPosition >= prevPoint.x)) ||
          ((bounds.getMaxY () >= HEIGHT) && (yPosition >= prevPoint.y)))
      {
         setCursor (DEFAULT_CURSOR);
      }
      else if (flags == MouseEvent.BUTTON1_MASK)
      {
         //NULLPOINTEXCEPTION SYT
         DataFlowComponent dfc = selectedComponent.getDataFlowComponent ();
         //moveing all components  SYT
         if (selectAllMode && (MOVING_ALL))
         {
            handlesVector.removeAllElements ();
            bounds = (Rectangle) ((DisplayComponent) displayComponentVector
             .elementAt (0)).getShape ().getBounds ();
            for (Enumeration enum = displayComponentVector.elements ()
                                            ; enum.hasMoreElements ();)
            {
               selectedComponent = (DisplayComponent) enum.nextElement ();
               dfc = selectedComponent.getDataFlowComponent ();
               dfc.moveTo (xPosition - prevPoint.x, yPosition - prevPoint.y);
               handlesVector.addAll (selectedComponent.getHandles ());
               handlesVector.addAll (selectedComponent
                .getStringHandles (selectedComponent.getLabelShapeBounds ()));
               handlesVector.addAll (selectedComponent
                .getStringHandles (selectedComponent.getMetShapeBounds ()));
               selectedComponent.update ();
               bounds = (Rectangle) bounds
                .createUnion (selectedComponent.getShape ().getBounds ());
            }

         }
         else if (RESIZING)
```

195

```
{
   // add 1/28/00 SYT
   draggedFlag=true;
   Rectangle2D.Double r2D = (Rectangle2D.Double) selectedComponent
                            .getShape ().getBounds2D ();
   r2D.setFrameFromDiagonal
                (new Point (xPosition, yPosition), diagonalPoint);
   ((Vertex) dfc).setX ((int) r2D.getCenterX ());
   ((Vertex) dfc).setY ((int) r2D.getCenterY ());
   ((Vertex) dfc).setWidth ((int) r2D.getWidth ());
   selectedComponent.update ();
   handlesVector = selectedComponent.getHandles ();
   //add   1/28/00 SYT
   parentFrame.setSaveRequired (true);
}
else if (MOVING_LABEL)
{
   // add 1/28/00 SYT
   draggedFlag=true;
   if (dfc instanceof External)
   {
      ((External) dfc).setLocation
                (xPosition - prevPoint.x, yPosition - prevPoint.y);
      selectedComponent.update ();
   }
   else
      dfc.setLabelOffset
                (xPosition - prevPoint.x, yPosition - prevPoint.y);
      handlesVector = selectedComponent
      .getStringHandles (selectedComponent.getLabelShapeBounds ());
}
else if (MOVING_MET)
{
   // add 1/28/00 SYT
   draggedFlag=true;
   dfc.setMetOffset
   (xPosition - prevPoint.x, yPosition - prevPoint.y);
   handlesVector = selectedComponent
   .getStringHandles (selectedComponent.getMetShapeBounds ());
}
else if (MOVING_COMPONENT)
{
   // add 1/28/00 SYT
   draggedFlag=true;
   if (dfc instanceof Vertex)
      ((Vertex) dfc).setLocation
                (xPosition - prevPoint.x, yPosition - prevPoint.y);
   else
      ((Edge) dfc).reShape (xPosition, yPosition);
   selectedComponent.update ();
   handlesVector = selectedComponent.getHandles ();
}
else
{
   if (selectedComponent.getLabelShapeBounds ()
                        .contains (xPosition, yPosition))
   {
      MOVING_LABEL = true;
      if (dfc instanceof External)
      {
         ((External) dfc).setLocation
                (xPosition - prevPoint.x, yPosition - prevPoint.y);
         selectedComponent.update ();
      }
      else
      dfc.setLabelOffset
                (xPosition - prevPoint.x, yPosition - prevPoint.y);
      handlesVector = selectedComponent
       .getStringHandles (selectedComponent.getLabelShapeBounds ());
```

196

```
                        parentFrame.setSaveRequired (true);
                    }
                    if (selectedComponent.getMetShapeBounds ()
                                            .contains (xPosition, yPosition))
                    {
                        MOVING_MET = true;
                        dfc.setMetOffset
                        (xPosition - prevPoint.x, yPosition - prevPoint.y);
                        handlesVector = selectedComponent
                         .getStringHandles (selectedComponent.getLabelShapeBounds ());
                        parentFrame.setSaveRequired (true);
                    }
                    else if (selectedComponent
                            .containsClickedPoint (xPosition, yPosition))
                    {
                        MOVING_COMPONENT = true;
                        if (dfc instanceof Vertex)
                           ((Vertex) dfc).setLocation
                                    (xPosition - prevPoint.x, yPosition - prevPoint.y);
                        else
                           ((Edge) dfc).reShape (xPosition, yPosition);
                        //selectedComponent.update ();
                        handlesVector = selectedComponent.getHandles ();
                        parentFrame.setSaveRequired (true);
                    }
                    //setCursor (MOVE_CURSOR);
                }
//clearAllComponentsFromScreen (null);
                paint (getGraphics ());
                setCursor (MOVE_CURSOR);
                prevPoint.setLocation (xPosition, yPosition);
            }
        }
    }
    //add 12/28/99 SYT
    /**
     *true if (x,y) in any rectangle of elements.
     *set the diagonalPoint.
     */
    public boolean isHoldingHandle (int x, int y)
    {
        boolean flag = false;
        Rectangle2D r2D;
        for (Enumeration e = handlesVector.elements (); e.hasMoreElements ();)
        {
            r2D = (Rectangle2D) e.nextElement ();
            if (r2D.contains (x, y))
            {
                diagonalPoint = getDiagonalPoint (r2D);
                flag = true;
            }
        }
        return flag;
    }

    public Point2D getDiagonalPoint (Rectangle2D rect)
    {
        Point2D p = new Point2D.Double ();
        int w = (int) rect.getWidth () / 2;
        Rectangle2D r2D = (Rectangle2D) selectedComponent
                                            .getShape ().getBounds ();
        if (rect.getMaxX () >= r2D.getMaxX ()
           && rect.getMaxY () >= r2D.getMaxY ())
           p.setLocation (r2D.getMinX () + w, r2D.getMinY () + w);
        else if (rect.getMaxX () >= r2D.getMaxX ()
           && rect.getMinY () <= r2D.getMinY ())
           p.setLocation (r2D.getMinX () + w, r2D.getMaxY () - w);
        else if (rect.getMinX () <= r2D.getMinX ()
           && rect.getMinY () <= r2D.getMinY ())
```

197

```
            p.setLocation (r2D.getMaxX () - w, r2D.getMaxY () - w);
        else if (rect.getMinX () <= r2D.getMinX ()
            && rect.getMaxY () >= r2D.getMaxY ())
            p.setLocation (r2D.getMaxX () - w, r2D.getMinY () + w);
        return p;
    }


    /**
     * Handles the event that occurs when the mouse is moved on this panel
     *
     * @param e The MouseEvent that occurs.
     */
    public void mouseMoved (MouseEvent e)
    {
        int xPosition = e.getX ();
        int yPosition = e.getY ();
        if (selectMode)
        {
            setCursor (DEFAULT_CURSOR);
            DisplayComponent dc;
            for (Enumeration enum = displayComponentVector.elements ()
                                              ; enum.hasMoreElements ();)
            {
                dc = (DisplayComponent) enum.nextElement ();
                if  (dc.containsClickedPoint (xPosition, yPosition) ||
                      dc.getLabelShapeBounds ().contains (xPosition, yPosition) ||
                      (dc.getMetShapeBounds ().contains (xPosition, yPosition)))
                {
                    if (dc.equals (selectedComponent)
                              && (MOVING_COMPONENT || MOVING_LABEL || MOVING_MET))
                        setCursor (MOVE_CURSOR);
                    else
                        setCursor (HAND_CURSOR);
                }
            }
        }
        //if (IS_COLLECTING_POINTS) {
        //    rubberBandLine (prevPoint.x, prevPoint.y);
        //    rubberBandLine (xPosition, yPosition);
        //}
    }

    public void actionPerformed (ActionEvent e)
    {

        if (e.getSource () == popupMenu.getDecomposeMenuItem ())
        {
            decompose();

        }
        if (e.getSource () == popupMenu.getFontMenuItem ())
        {
            String selected = (String) JOptionPane.showInputDialog
                              (parentFrame, "Select Font : ", "Font Selection"
                               , JOptionPane.INFORMATION_MESSAGE, null
                               , FontConstants.FONT_NAMES
                               ,FontConstants.FONT_NAMES [0]);
            if (selected != null)
            {
                int fontIndex = 0;
                for (int ix = 0; ix < FontConstants.FONT_NAMES.length; ix++)
                {
                    if (FontConstants.FONT_NAMES [ix].equals (selected))
                        fontIndex = ix;
                }
                selectedComponent.getDataFlowComponent ()
                                    .setLabelFontIndex (fontIndex + 1);
                selectedComponent.getDataFlowComponent ()
                                    .setMetFontIndex (fontIndex + 1);
```

198

```java
            selectedComponent.setLabelShape ((Graphics2D) getGraphics ());
            selectedComponent.setMetShape ((Graphics2D) getGraphics ());
//clearAllComponentsFromScreen ((Graphics2D) getGraphics ());
            paint (getGraphics ());
        }
      }
      else if (e.getSource () == popupMenu.getColorMenuItem ())
      {
        String selected = (String) JOptionPane
                             .showInputDialog (parentFrame, "Select color : "
                             , "Open", JOptionPane.INFORMATION_MESSAGE, null
                             , ColorConstants.COLOR_NAMES
                             , ColorConstants.COLOR_NAMES [0]);
        if (selected != null)
        {
            int colorIndex = 0;
            for (int ix = 0; ix < ColorConstants.COLOR_NAMES.length; ix++)
            {
               if (ColorConstants.COLOR_NAMES [ix].equals (selected))
                  colorIndex = ix;
            }
            ((Vertex) selectedComponent.getDataFlowComponent ())
                                       .setColor (colorIndex + 1);
//clearAllComponentsFromScreen ((Graphics2D) getGraphics ());
            paint (getGraphics ());
        }
      }
      else if (e.getSource () == popupMenu.getDeleteMenuItem ())
      {
         deleteSelectedComponent ();
      }
      else if (e.getSource () == popupMenu.getPropMenuItem ())
      {
         showProperties (selectedComponent);
      }
    }
  }
  //12/11/99 SYT
  /**
   * Editor class will use this for solving key board
   * event can not be accessed problem.
   */
  public void deleteSelectedComponent ()
  {
     //prevent null pointer exception  7/28/00 SYT
     if(selectedComponent != null)
     {
     if(selectedComponent.getDataFlowComponent().getChildCount()!=0 )
     {
       //add 2/7/00 SYT
       new DeleteDialog(this,false);
     }
     else
     {
       deleteSelected();
     }
     }
  }
   // add 2/7/00 SYT
  /**
   * delete seleted component and it's children
   */
  public void deleteSelected()
  {
     DefaultMutableTreeNode selectedParent =
     (DefaultMutableTreeNode)(((DefaultMutableTreeNode)
     (selectedComponent.getDataFlowComponent ())).getParent());
     if (!(selectedComponent.getDataFlowComponent () instanceof External))
     {
        pushUndoVector();
```

199

```
          //delete it and it's childs. SYT
        selectedComponent.delete ();

        displayComponentVector.removeElement (selectedComponent);

          //error. 1/6/00 SYT
          //parentFrame.getTreePanel ().removeDfc (selectedComponent
          //.getDataFlowComponent ());
        parentFrame.getTreePanel ().removeDfc();
          // This takes care of unremoved streams
        setParentVertex (parentVertex, null);
        parentFrame.setSaveRequired (true);
      }
      //change on 2/28/00 SYT
     //set path of the selected component
      //debug : null pointer exception.  6/12/00 SYT
      if(selectedParent != null )
      {
         if(selectedParent.isLeaf())
         {
           //add if statement to debug null pointer exception 6/10/00 SYT
           if( (DefaultMutableTreeNode)selectedParent
                         .getParent() != null)
           {
             TreeNode [] obj= ((DefaultMutableTreeNode)selectedParent
                         .getParent()).getPath();
             //add 2/12/00 SYT
             //following lines will make labe marker disapear
             TreePath TP = new TreePath(obj);
             parentFrame.getTreePanel().setSelectionPath(TP);
             //ensure selected path visible
             parentFrame.getTreePanel().expandPath(TP);
           }
         }
         else
         {
           TreeNode [] obj= selectedParent.getPath();
           //add 2/12/00 SYT
           //following lines will make label marker disapear
           TreePath TP = new TreePath(obj);
           parentFrame.getTreePanel().setSelectionPath(TP);
           //ensure selected path visible
           parentFrame.getTreePanel().expandPath(TP);
         }
      }
}

 //add  SYT 12/31/99
/**
 *  clear all tree node from root.
 */
 public void deleteAllSelectedComponents ()
 {
    //add 2/9/00 SYT
   boolean  hasChildenFlag = false;
     //add 2/7/00 SYT
   for(Enumeration e = parentVertex.children() ; e.hasMoreElements() ;)
   {
      if( ((DefaultMutableTreeNode)(e.nextElement())).getChildCount() != 0)
      hasChildenFlag=true;
   }
   //add 2/9/00 SYT
   if(hasChildenFlag)
   {
      new DeleteDialog(this,true);
   }
   else
   {
      deleteAllSelected();
```

```java
        }

    }
    // add 2/7/00 SYT
  /**
   * for clear all tree nodes
   * and their children from root.
   */
  public void deleteAllSelected ()
  {
      pushUndoVector();
      DataFlowComponent  DFC =null;
      if(!handlesVector.isEmpty())
      {
          handlesVector.removeAllElements ();
      }
      if (!displayComponentVector.isEmpty ())
      {
         while (parentVertex.getChildCount() != 0)
         {
            ((DataFlowComponent)(parentVertex.getChildAt(0)) ).delete();
         }

         displayComponentVector.removeAllElements();
      }
      selectAllMode=false;
      //add 1/13/00 SYT
      setSelectMode (true);
      parentFrame.setSaveRequired (true);
//clearAllComponentsFromScreen(null);

      parentFrame.getTreePanel ().removeDfc();
      // setParentVertex (parentVertex, null);
      paint (getGraphics ());

   }

   public Vertex getParentVertex ()
   {
      return parentVertex;
   }

   public void setSelectAllMode (boolean b)
   {
      selectAllMode = b;
      if (selectAllMode)
      {  selectAllComponents ();
         //add 1/13/00 SYT
         setSelectMode(true);
      }
   }

   //add SYT 12/31/11
   /**
    * current select mode
    */
   public boolean getSelectAllMode()
   {
      return selectAllMode;
   }

   public void selectAllComponents ()
   {
      DisplayComponent dc = null;
      handlesVector.removeAllElements ();
      if (!displayComponentVector.isEmpty ())
      {
         bounds = (Rectangle) ((DisplayComponent) displayComponentVector.elementAt
(0)).getShape ().getBounds ();
```

```
        }
        for (Enumeration enum = displayComponentVector.elements (); enum.hasMoreElements
();;)
        {
           dc = (DisplayComponent) enum.nextElement ();
           handlesVector.addAll (dc.getHandles ());
           handlesVector.addAll (dc.getStringHandles (dc.getLabelShapeBounds ()));
           handlesVector.addAll (dc.getStringHandles (dc.getMetShapeBounds ()));
            bounds = (Rectangle) bounds.createUnion (dc.getShape ().getBounds ());
        }
        selectedComponent = dc;
        paint (getGraphics ());
    }

    public void setSelectionDefault (boolean b)
    {
        selectionDefault = b;
    }

    public void setCurrentColor (int colorIndex)
    {
        currentColor = colorIndex;
    }

    public void setCurrentFont (int fontIndex)
    {
        currentFont = fontIndex;
    }

    protected void rubberBandLine (int x, int y)
    {
        Point last = (Point) ((Edge) currentEdge.getDataFlowComponent ()).getPoints
().lastElement ();
        Graphics g = getGraphics ();
        g.setColor (new Color (128, 128, 128));
        g.setXORMode (Color.white);
        g.drawLine (last.x, last.x, x, y);
        g.setPaintMode ();
        g.setColor (Color.black);
    }
    //add ket evet 2/3/00 SYT
    public void keyPressed(KeyEvent e)
    {
        if(e.getKeyChar()==e.VK_ESCAPE)
        {
          clearnDrawStream();
        }
    }

    public void keyTyped(KeyEvent e)
    {
    }

    public void keyReleased(KeyEvent e)
    {

    }

    //* added 02/07/01 to double buffer the drawPanel in the splitpane of editor--jam
    public void update ( Graphics g )
    {
        paint(g);

    }

} // End of the class DrawPanel.
```

```
package caps.CAPSMain;

import java.awt.*;
import javax.swing.*;
import javax.swing.filechooser.FileSystemView;
import java.io.*;
import java.io.File;
// 10/18/00 MTS
// replace caps.Builder by caps.PsdlBuilder and caps.TypeBuilder
// import caps.Builder.PsdlBuilder;
import caps.PsdlBuilder.PsdlBuilder;
import caps.TypeBuilder.TypeBuilder;
import caps.Psdl.Vertex;
import caps.Psdl.DataTypes;
import caps.GraphEditor.Editor;
import caps.FileManager.Manager;
import caps.AdaFileEditor.TextEditor;
import java.awt.event.*;
import java.util.Vector;
import java.util.Enumeration;
import java.io.IOException;
//import caps.images.jar;

/**
  * The main CAPS window.
  *
  * @author Shen-Yi Tao
  * @version 1.1
  */

/**
  * Changes
  *    7-9-99 MTS
  *    changed the capsLabel from "Heterogeneous System Intergrator"
  *    to "Heterogeneous Systems Integrator"
  *
  *    7-9-99 MTS
  *    add 3 private attributes: protoHome, protoName, protoVersion
  *    add 3 methods to update the private attrbutes
  *
  *    7-9-99 MTS
  *    added translatePrototype()
  *
  *    7-12-99 MTS
  *    added private attribute": CAPSJavaHome
  *
  *    7-12-99 MTS
  *    added schedulePrototype(), compilePrototype(),
  *    executePrototype()
  *
  *    11/3/00 MTS
  *    change the call in translatePrototype() from
  *    translate.script to translate.sh
  *    to work around the standard output redirection
  *    problem in run.exec()
  *
  *    change the call in schedulePrototype() from
  *    make.script to make.sh
  *    to work around the standard output redirection
  *    problem in run.exec()
  *
  *    change the call in compilePrototype() from
  *    compile.script to compile.sh
  *    to work around the standard output redirection
  *    problem in run.exec()
  *
  */
```

```java
public class CAPSMainWindow extends JFrame
{

    /**
      * The width of the frame.
      */
    private final int WIDTH = 400;

    /**
      * The height of the frame.
      */
    private final int HEIGHT = 250;

    /**
      * The File that contains the PSDL prototype.
      */
    private File prototype;
    //add 8/26/00 SYT
    /**
      * The Folder that contains the PSDL prototype.
      */
    private File adaTemplet;

    /**
      * 7-9-99 MTS
      * add private attribute to hold protoHome
      * default protoHome = $HOME in UNIX
      * and = C:\Windows in Windows
      */
    private static String protoHome;

    /**
      * 7-9-99 MTS
      * add private attribute to hold protoName
      */
    private static String protoName;

    /**
      * 7-9-99 MTS
      * add private attribute to hold protoVersion
      */
    private static String protoVersion;

     /**
      * 7-24-01
      * add private attribute to hold userName
      */

    private static String userName;

    /**
      * 7-24-01
      * add private attribute to hold projectName
      */
    private static String projectName;

    /**
      * 7-24-01
      * add private attribute to hold projectVersion
      */
    private static String projectVersion;

    /**
      * 7-12-99 MTS
      * add private attribute to hold CAPSJavaHome
      */
    private static String serverUrl;

    /**
```

```
      * 13 Aug 01 JAM
      * add private attribute String of appserver url
      */
    private static String CAPSJavaHome;

    /**
      * The Vector that holds references to the open prototypes
      */
    private static Vector openPrototypes;

    /**
     * Buffers for file copy
     */

    static final int BUFF_SIZE = 100000;
    static final byte[] buffer = new byte[BUFF_SIZE];

    /**
      * The constructor for this class.
      */
    public CAPSMainWindow ( String user, String project, String version )
    {
        super ("Web-CAPS");       // The title of the frame.

        userName = user;
        protoName = project;
        protoVersion = version;
        System.out.println( "User:  "+userName+"  Project:  "+protoName+"  Version:
"+protoVersion );

        prototype = null;
        adaTemplet = null;

        openPrototypes = new Vector (0, 2);

        serverUrl = "http://seaotter.cs.nps.navy.mil:8080/CapsWeb";

        initialize ();


    }

    /**
      * Initializes the CAPS main window.
      */
    public void initialize ()
    {
        setDefaultCloseOperation(WindowConstants.DO_NOTHING_ON_CLOSE);
        addWindowListener (new ExitCAPSMain (this));

        /**
          * Places the frame in the upper-right corner of the screen
          */
         Dimension screenSize = Toolkit.getDefaultToolkit().getScreenSize();

            setLocation(screenSize.width - (WIDTH + WIDTH / 2), HEIGHT / 2);

        setResizable (false);

        setJMenuBar (new CAPSMainMenuBar (this));

        BorderLayout innerLayout = new BorderLayout ();
        innerLayout.setVgap (1);


        JPanel initPanel = new JPanel(innerLayout);
        initPanel.setBorder (BorderFactory.createLoweredBevelBorder ());
        initPanel.setBackground(Color.gray);
```

205

```
      JPanel panel = new JPanel ();
      panel.setBorder (BorderFactory.createRaisedBevelBorder ());
      JPanel labelPanel = new JPanel();
      labelPanel.setBorder (BorderFactory.createRaisedBevelBorder ());
      // MTS 7/9/99 Change the capsLabel in the following state
      // from "Heterogeneous System Integrator"
      // to "Heterogeneous Systems Integrator"

      JLabel capsLabel = new JLabel ("Web-CAPS Prototype Developer" );
      capsLabel.setFont (new Font ("Courier", Font.BOLD, 17));
      JLabel projectLabel = new JLabel ("User:  "+userName+"    Project:  "+protoName+"
Version:  "+protoVersion);
      projectLabel.setFont (new Font ("Courier", Font.BOLD, 12));
      projectLabel.setAlignmentX(CENTER_ALIGNMENT);
      // MTS 7/12/99 Add code to initialize CAPSJavaHome in order for
      // program to locate caps/Images correctly

      // The system property for the CAPS classes directory.
      CAPSJavaHome = System.getProperty ("CAPSJavaHome");
      if (CAPSJavaHome == null)
      {
          CAPSJavaHome = ".";
      }
      //System.out.println ("CAPSJavaHome = " + CAPSJavaHome);

      //JLabel imageLabel = new JLabel (new ImageIcon ("caps/Images/caps.gif"));

      //used to fix image icon problems with jar file implementation
      ClassLoader cl = this.getClass().getClassLoader();
      Icon capsIcon = new ImageIcon(cl.getResource("caps/Images/caps.gif"));
      JLabel imageLabel = new JLabel(capsIcon);
      //JLabel imageLabel =
        // new JLabel (new ImageIcon (CAPSJavaHome + "/caps/Images/caps.gif"));


      panel.add (Box.createHorizontalStrut (5));
      panel.add (imageLabel);
      panel.add (Box.createHorizontalStrut (5));
      panel.add (capsLabel);
      panel.add (Box.createHorizontalStrut (5));

      labelPanel.add(projectLabel);

      initPanel.add(panel,BorderLayout.NORTH);

      initPanel.add(labelPanel,BorderLayout.SOUTH);

      getContentPane ().add (initPanel);

      pack ();

      setVisible (true);
   }

   /**
     * Sets the prototype file to the argument.
     *
     * @param f The File that contains the PSDL prototype.
     */
   public void setPrototype (File f)
   {
      prototype = f;
      // MTS 7-9-99 added the following debug statement
      //System.out.println ("Prototype Name = " + prototype.getName());
      //System.out.println ("Prototype Name Length = "
      //                    + (prototype.getName()).length());
   }

   public File getPrototype ()
```

```
{
    return prototype;

}

 //add 8/26/00 SYT
/**
  * Sets the add templet file to the argument.
  *
  * @param t The File to create the  ada templet.
  */
public void setAdaTemplet(File t)
{
    adaTemplet = t;
}

/**
  * Gets the add templet file to the argument.
  *
  * @param t The File to create the  ada templet.
  */
public File getAdaTemplet()
{
    return adaTemplet;
}

/**
  * Sets the prototype home directory name to the argument.
  *
  * @param s The string that contains the prototype home dir.
  */
public void setProtoHome (String s)
{
    protoHome = s;
    // debug statement
    System.out.println ("Prototype Home = " + s);
}

/**
  * Gets the prototype home directory name to the argument.
  *
  * @param s The string that contains the prototype home dir.
  */
public String getProtoHome ()
{
    return protoHome;

}

/**
  * Sets the prototype name to the argument.
  *
  * @param s The string that contains the prototype name.
  */
public void setProtoName (String s)
{
    protoName = s;
    // debug statement
    System.out.println ("Prototype Name = " + s);
}

/**
  * Gets the prototype name to the argument.
  *
  * @param s The string that contains the prototype name.
  */
public String getProtoName ()
{
    return protoName;
```

```java
}

/**
  * Sets the prototype version to the argument.
  *
  * @param s The string that contains the prototype name.
  */
public void setProtoVersion (String s)
{
    protoVersion = s;
    // debug statement
    System.out.println ("Prototype Version = " + s);
}

/**
  * Gets the prototype version to the argument.
  *
  * @param s The string that contains the prototype name.
  */
public String getProtoVersion ()
{
    return protoVersion;

}

/**
  * Returns the vector that holds the open prototype files.
  *
  * @return the vector that contains the open prototype files.
  */
public Vector getOpenPrototypes ()
{
    return openPrototypes;
}

/**
  * Opens the graphics editor to edit a prototype.
  * Translate file to symbols      SYT
  */
public void editPrototype ()
{
    if (prototype == null)
    {           // No prototype is selected to open
      JOptionPane.showMessageDialog (this,"No prototype is selected to edit."
      ,"Error Message", JOptionPane.ERROR_MESSAGE);
    }
    else if (!isPrototypeChanged ())
    {          // Attempt to edit the same prototype.
      JOptionPane.showMessageDialog (this
      , new String ("Prototype " + prototype.getName () +" is already open.")
      , "Error Message", JOptionPane.ERROR_MESSAGE);
    }

    else
    {  //read and translate prototype from text to graphics. SYT
      PsdlBuilder.disable_tracing ();        // Disable debug messages
      Vertex root = null;
      root = PsdlBuilder.buildPrototype (prototype);
      if (root == null)
      {
         // If this is a new prototype ,Prototype name
         // is the same as  the file name.  SYT
         root = new Vertex (0, 0, null, false);
         String name = prototype.getName ();
         root.setLabel (name.substring (0, name.length () - 5));
      }
      DataTypes types = TypeBuilder.buildType (prototype);
```

208

```
         Editor e = new Editor (prototype, adaTemplet, root, types);
         new Thread (e).start ();
         openPrototypes.addElement (e);
      }
   }

   /**
     * Opens the File Manager for local and server project files
     * JAM 4 Aug 01
     */
   public void manageFiles(){
         Manager m = new Manager ( adaTemplet, serverUrl, userName, protoName,
protoVersion );
         new Thread (m).start ();

      }

   public void openAdaEditor(){
         TextEditor t = new TextEditor();
         new Thread (t).start();

      }


      public void addSupportFiles(){

         File[] filesToCopy;//array of files to copy
         String copyTo = "";//String to hold copy to directory

         File temp = getAdaTemplet();

             try{
                 copyTo = temp.getCanonicalPath();
             }
             catch(IOException f){

                 System.out.println(f);
             }


         JFileChooser fileChooser = new JFileChooser();
         fileChooser.setFileSelectionMode(JFileChooser.FILES_ONLY);
         fileChooser.setMultiSelectionEnabled(true);
         fileChooser.setApproveButtonText("Copy");
         fileChooser.setDialogTitle("Copy support files to prototype folder");

         int result = fileChooser.showOpenDialog(this);
         if( result == JFileChooser.CANCEL_OPTION ){

             return;

         }
         if(result == JFileChooser.APPROVE_OPTION ){

             filesToCopy = fileChooser.getSelectedFiles();

             for(int i = 0; i < filesToCopy.length; i++){

                 String from = "";
                 String to   = "";

                 try{
                     from = filesToCopy[i].getCanonicalPath();
                     to   = filesToCopy[i].getName();
                 }
                 catch(IOException e){
                     System.out.println(e);
                 }
```

```
                try{
                    copy( from, copyTo+File.separator+to );
                }
                catch(IOException f){
                    System.out.println(f);

                }

        }

    }


    /**
      * Checks whether or not the current prototype file is already used by
      * a PSDL Editor.
      *
      * @return true if one of the open prototypes is the same as the current
      *  prototype file.
      */
    public boolean isPrototypeChanged ()
    {
        for (Enumeration enum = openPrototypes.elements ()
                                        ; enum.hasMoreElements ();)
        {
            Editor e = (Editor) enum.nextElement ();
            if (prototype.equals (e.getPrototypeFile ()))
                return false;
        }
        return true;
    }

    /**
      * Removes one element from the openPrototypes vector.
      *
      * @param e the editor that is going to be removed from the vector.
      */
    public static void removeEditor (Editor e)
    {
        openPrototypes.removeElement (e);
    }

    /**
      * Checks if the status of any of the open prototypes is 'saveRequired'.
      * Prompts the user to save the prototype.
      *
      * @return true if none of the prototypes need saving.
      */
    public boolean isOpenPrototypeSaved ()
    {
        boolean flag = true;
        Editor e;
        label :
        for (Enumeration enum = openPrototypes.elements ()
                            ;enum.hasMoreElements ();)
        {
            e = (Editor) enum.nextElement ();
            if (e.isSaveRequired ())
            {
                int ix = JOptionPane.showConfirmDialog
                            (this, new String ("Save changes to the prototype " +
                                            e.getRoot ().getLabel () + "?"));
                if (ix == JOptionPane.CANCEL_OPTION)
                {
```

```
            flag = false;
            break label;
          }
          else if (ix == JOptionPane.YES_OPTION)
              e.savePrototype ();
      }
    }
    return flag;
}

// MTS 7/9/99 added procedure translatePrototype()
/**
 * printing run time project info.
 */
public void translatePrototype()
{
      // MTS 11/3/00 change the call from translate.script to translate.sh
      String command = "translate.sh " + protoHome + " " + protoName + " "
                        + protoVersion;
      System.out.println (command);
      try
      {
        Runtime run = Runtime.getRuntime ();
        run.exec (command);
      } catch (IOException ex)
      {
        System.out.println (ex);
      }
}

// MTS 7/12/99 added procedure schedulePrototype()
public void schedulePrototype()
{
      // MTS 11/3/00 change the call from make.script to make.sh
      String command = "make.sh " + protoHome + " " + protoName + " "
                        + protoVersion;
      System.out.println (command);
      try
      {
        Runtime run = Runtime.getRuntime ();
        run.exec (command);
      } catch (IOException ex)
      {
        System.out.println (ex);
      }
}

// MTS 7/12/99 added procedure schedulePrototype()
public void compilePrototype()
{
      // MTS 11/3/00 change the call from command.script to command.sh
      String command = "compile.sh " + protoHome + " " + protoName + " "
                        + protoVersion;
      System.out.println (command);
      try
      {
        Runtime run = Runtime.getRuntime ();
        run.exec (command);
      }
      catch (IOException ex)
      {
        System.out.println (ex);
      }
}

// MTS 7/12/99 added procedure schedulePrototype()
public void executePrototype()
{
      String command = "execute.script " + protoHome + " " + protoName + " "
```

```
                            + protoVersion;
          System.out.println (command);
          try
          {
             Runtime run = Runtime.getRuntime ();
             run.exec (command);
          }
          catch (IOException ex)
          {
             System.out.println (ex);
          }
      }


    /**
     * Copy file to local directory for inclusion in project
     * Added 08/22/01 JAM
     */

    public static void copy(String from, String to)throws IOException{

        InputStream in = null;
        OutputStream out = null;
        try{
            in = new FileInputStream(from);
            out = new FileOutputStream(to);
            while(true){
                synchronized (buffer){
                    int amountRead = in.read(buffer);
                    if(amountRead == -1){
                        break;
                    }
                    out.write(buffer,0,amountRead);

                }
            }
        }finally{
            if(in != null){
                in.close();
            }
            if(out != null){
                out.close();
            }
        }
    }//end of copy




} // End of the class CAPSMainWindow
```

```java
package caps.CAPSMain;

import javax.swing.JMenuBar;

/**
  * The menubar of the main CAPS window.
  *
  * @author Ilker DURANLIOGLU
  * @version
  */

/**
  * Changes:
  *    7-9-99 MTS
  *    added the owner parameter to the PrototypeMenu constructor call
  *    in the CAPSMainMenuBar method
  */

public class CAPSMainMenuBar extends JMenuBar
{

   /**
     * The constructor for this class.
     *
     * @param owner The parent class which has declared this menubar.
     */
   public CAPSMainMenuBar (CAPSMainWindow owner)
   {
      super ();

      // Add the menus
      add (new PrototypeMenu (owner));
      add (new EditMenu (owner));
      //add (new DatabasesMenu ());

      // 7-9-99 MTS
      // added owner parameter to the following constructor call
      //add (new ExecSupportMenu (owner));
      add (new HelpMenu ());
   }


} // End of the class CAPSMAinMenuBar
```

```java
package caps;

import caps.CAPSMain.*;

/**
  * The driver program for CAPS.
  *
  * @author Ilker DURANLIOGLU
  * @version
  */
public class Caps
{


  /**
     * The constructor for this class.
     *
     * @param args[] The command line parameters.
     * (No command line parameter is necessary for this program.)
     */
   public static void main (String args [])
    {


       String user = String.valueOf(args[0]);
       String project = String.valueOf(args[1]);
       String version = String.valueOf(args[2]);

       CAPSMainWindow main = new CAPSMainWindow ( user, project, version );


    }


} // End of the class Caps
```

```java
package caps;

import caps.CAPSMain.*;

/**
  * The driver program for CAPS.
  *
  * @author Ilker DURANLIOGLU
  * @version
  */
public class Caps
{


  /**
     * The constructor for this class.
     *
     * @param args[] The command line parameters.
     * (No command line parameter is necessary for this program.)
     */
   public static void main (String args [])
    {


       String user = String.valueOf(args[0]);
       String project = String.valueOf(args[1]);
       String version = String.valueOf(args[2]);

       CAPSMainWindow main = new CAPSMainWindow ( user, project, version );


    }


} // End of the class Caps
```

THIS PAGE INTENTIONALLY LEFT BLANK

## APPENDIX C APPLICATION SERVER SETUP


The application server chosen was a UNIX version of Tomcat 3.2.3. Tomcat fully supports Java Servlets. The application server has two applications, the administrator interface and the user interface. No special configuration was necessary for the server. The server was configured as directed by the users manual. The mime type for the Java Networking Language Protocal had to be added to the mime types of the server. This addition allows for the invocation of Java Web Start on the client system.

The application server also houses the file system for the user workspaces. Each user has a workspace on the server were all project files are saved. The access to the file system is synchronized using Java servlets and the database management system. See the servlet source code for specifics.

Once the Java Servlets have been compiled using the Java compiler and Servlet API they are saved in the application folder. When the server is initiated it automatically initiates all servlets for the applications on the server and they are available for users to use. The standard startup script is used to start the server: "startup.sh"

THIS PAGE INTENTIONALLY LEFT BLANK

# APPENDIX D CLIENT SYSTEM SETUP

Software Required:

- Operating System:  Microsoft Windows, UNIX, Linux

- Java Runtime Environment (JRE): 1.2.2 and 1.3.1

- Java Web Start 1.0 or above.

- Web browser: Microsoft Internet Explorer 5.0 or Netscape Navigator 4.X.

The above listed software is what is required to execute SEAS on a local client system.  The operating systems listed all have versions of JRE, Java Web Start, and web browsers that are capable of executing the system. The need for two JRE's is required because the compiler, JGNAT 1.0, used to compile the final project creates an executable Java Archive File (JAR) compatible only with Java 1.2.2.  The remainder of the Java Components are developed in Java 1.3.1.

When setting up Java Web Start it is recommended to turn the Java console "on" in the preferences.  This will allow the user to view standard input and output from both the prototypes executed as well as the PSDL Editor.

THIS PAGE INTENTIONALLY LEFT BLANK

# APPENDIX E DATABASE MANAGEMENT SYSTEM SETUP

The database management system (DBMS) utilized for SEAS was the open source relational DBMS called MySql. MySql was set up as a service on a computer running the Windows 2000 operating system.  As a service the DBMS would be available for SEAS as long as the computer was powered up.  The DBMS is accessed via the SEAS administrative and user interface servlets using Java Database Connectivity (JDBC).  Each servlet that was required to query or update the DBMS was equipped with a method to connect to the DBMS. See the servlet source code for specifics.

THIS PAGE INTENTIONALLY LEFT BLANK

# LIST OF REFERENCES

[1]  Luqi, V. Berzins and R. Yeah, "A Prototyping Language for Real Time Software", *IEEE Transactions on Software Engineering,* Vol. 14, No. 10, October 1988, pp. 1409-1423.

[2]  Luqi and M. Ketabchi, "A Computer Aided Prototyping System", *IEEE Software*, pp.66-72, March 1988.

[3]  Luqi, "System Engineering and Computer-Aided Prototyping", *Journal of Systems Integration, Special Issue on Computer Aided Prototyping*, Vol. 6, 1996, pp. 15-17.

[4] I. Duranlioglu, *Implementation of a Portable PSDL Editor for Heterogeneous Systems Integrator*, Master's Thesis, Naval Postgraduate School, Monterey, CA., March 1999.

[5]  G. Kreeger, *Requirements Analysis and Design of a Distributed Architecture for the Computer Aided Prototyping System*, Master's Thesis, Naval Postgraduate School, Monterey, CA., September 1999.

[6]  Shen-Yi Tao, *Design and Implementation of a Platform Independent Prototype Specification Editor*, Master's Thesis, Naval Postgraduate School, Monterey, CA., September 2000.

[7]  J. Conallen, "Modeling Web Application Architectures with UML", *Communications of the ACM*, Vol. 42, No. 10, October 1999.

[8]  D. Leffingwell, D. Widrig, *Managing Software Requirements A Unified Approach*, Addison-Wesley, 2000.

[9]  M. Fowler, *UML Distilled Second Edition*, Addison-Wesley, 2000.

[10]   Java 2 SDK, Standard Edition Documentation,
http://java.sun.com/products/jdk/1.3.1/docs/index.html,
August 2001.

[11]   Java Servlet 2.2 Documentation,
http://java.sun.com/products/servlet/2.2/javadoc/index.html
, August 2001.

[12]   Java Web Start Documentation,
http://java.sun.com/products/javawebstart/index.html,
August 2001.

[13]   Jakarta-Tomcat 3.2.3 Documentation,
http://jakarta.apache.org/, August 2001.

[14]   Carnegie-Mellon University Software Engineering
Institute, Software Technology Review, *Three Tier Software
Architectures*,
http://www.sei.cmu.edu/str/descriptions/threetier_body.html
, August 2001.

[15]   J. Jaworski, *Mastering JavaScript*, Sybex, 1997.

[16]   JDBC Data Access,
http://java.sun.com/products/jdbc/index.html, August 2001.

# INITIAL DISTRIBUTION LIST

1.  Defense Technical Information Center
    Ft. Belvoir, VA

2.  Dudley Knox Library
    Naval Postgraduate School
    Monterey, CA

3.  Professor Man-Tak Shing, Code CS/Sh
    Computer Science Department
    Naval Postgraduate School
    Monterey, CA 93943-5100

4.  Professor Luqi, Code CS/Lq
    Computer Science Department
    Naval Postgraduate School
    Monterey, CA 93943-5100

5.  LtCmdr Chris Eagle, Chairman, Code CS
    Naval Postgraduate School
    Monterey, California

6.  Marine Corps Representative
    Naval Postgraduate School
    Monterey, California

7.  Director, Training and Education, MCCDC, Code C46
    Quantico, Virginia

8.  Director, Marine Corps Research Center
    MCCDC,Code C40RC
    Quantico, Virginia

9.    Marine Corps Tactical Systems Support Activity
      (Attn: Operations Officer)
      Camp Pendleton, California